IMPLEMENTASI SISTEM INDIKASI KEBAKARAN MENGGUNAKAN MIKROKONTROLER DALAM *SMARTHOME*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Lazuardy Muhammad
NIM: 115060807111147

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI SISTEM INDIKASI KEBAKARAN MENGGUNAKAN MIKROKONTROLER DALAM *SMARTHOME*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Lazuardy Muhammad
NIM: 115060807111147

Skripsi ini telah diuji dan dinyatakan lulus pada
3 Agustus 2018

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Dany Primanita Kartikasari, S.T., M.Kom	Eko Sakti Pramukantoro, S.Kom, M.Kom
NIP: 197711162005012003	NIK: 2011028608051001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 197105182003121001

ii

PENGUJI

• Penguji I / Ketua Majelis

Fariz Andri Bakthiar, S.T., M.Kom.

NIK. 85031406110028

• Ari Kusyanti, S.T, M.Sc

NIK. 2011028312282001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 31 Juli 2018

Lazuardy Muhammad

NIM: 115060807111147

NAMA LENGKAP : LAZUARDY MUHAMMAD
JENIS KELAMIN : LAKI-LAKI
TTL : BANYUWANGI, 7 JULI 1993
FAKULTAS : FAKULTAS ILMU KOMPUTER
UNIVERSITAS : UNIVERSITAS BRAWIJAYA MALANG
JURUSAN / PRODI : TEKNIK INFORMATIKA
NIM : 115060807111147
NO HP : 081232620846
ALAMAT DI MALANG : JL. KH. MALIK DALAM NO.39
AGAMA : ISLAM
ANAK KE : 1DARI 3 BERSAUDARA

RIWAYAT PENDIDIKAN

TK SABILILLAH MALANG 1998
MI KHADIJAH MALANG 2000
MTSN 1 MALANG 2006
SMAN 4 MALANG 2009

UCAPAN TERIMA KASIH

Puji syukur kehadirat Allah SWT karena atas rahmat dan hidayah-Nya, penulis dapat menyelesaikan skripsi dengan judul "**Implementasi Indikasi Sistem Kebakaran Menggunakan Mikrokontroler dalam *Smarthome* **".

Penyusunan skripsi ini tidak lepas dari bantuan semua pihak yang telah memberikan semangat, doa, bimbingan, kritik, serta saran. Maka dari itu penulis menyampaikan ucapan terimakasih kepada:

1. Seluruh keluarga yang telah membantu dan mendukung demi kelancaran pengerjaan skripsi ini.
2. Rekan-rekan yang telah memberikan dukungan baik secara moral, pemikiran maupun doa.
3. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya, Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika Universitas Brawijaya dan Bapak Agus Wahyu Widodo, S.T, M.Sc. selaku Ketua Prodi Teknik Informatika
4. Ibu Dany Primanita Kartikasari, S.T., M.Kom. selaku dosen pembimbing I yang telah membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini.
5. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat baik secara langsung maupun tidak langsung demi terselesaikannya tugas akhir ini.

Semoga skripsi ini dapat bermanfaat dan berguna bagi pembaca terutama mahasiswa Prodi Teknik Informatika Jurusan Teknik Informatika Universitas Brawijaya.

Malang, 31 Juli 2018

Lazuardy Muhammad

lazuzalazu@gmail.com

ABSTRAK

Dalam lingkungan perkotaan kebakaran merupakan peristiwa yang sering terjadi. Penyebab utama dari kebakaran dalam perkotaan adalah arus pendek dan kebocoran gas. Kebocoran gas pada rumah umumnya terjadi pada dapur. Maka dibutuhkan suatu sistem agar dapat membaca indikasi jika terjadi kebakaran. Dengan menggunakan *smarthome* perangkat perangkat dalam rumah dapat dikendalikan dan dimonitor menggunakan bantuan sensor serta mikrokontroler. Sistem pendeteksi indikasi kebakaran diimplementasikan dengan cara menghubungkan mikrokontroler dengan sensor sehingga sensor dapat mengirimkan data. Sensor yang digunakan dalam penelitian adalah sensor api sensor MQ-135 dan *buzzer*. Pertukaran data pada sistem menggunakan protokol MQTT. MQTT adalah protokol komunikasi pada IoT yang menerapkan *publish* dan *subscribe*. Sensor yang bertugas sebagai publisher adalah sensor api dan asap sedangkan sensor yang bertugas sebagai subscriber adalah *buzzer*. Protokol MQTT adalah protokol komunikasi mesin ke mesin, dapat beroperasi pada daya rendah dan dapat mengirim pesan dengan cepat. Sensor membaca data dari lingkungan, jika data yang dibaca melebihi parameter, maka akan menyebabkan *buzzer* berbunyi. Pengujian sistem dilakukan dengan cara menghitung waktu sistem dalam mengirim data dari sensor sampai menuju buzzer dan dapat menampilkan data secara lagsung dan grafik, pengaturan pada *buzzer* dan parameter untuk sensor. Hasil dari pengujian sistem dapat membaca, menampilkan data secara *realtime* serta dapat menampilkan data dalam bentuk grafik dan kecepatan pengiriman sensor dari awal sampai berbunyinya buzzer adalah 1 detik.

Kata kunci : Smarthome, MQTT, mikrokontroler dan Sensor

ABSTRACT

In urban environments fires are a frequent occurrence. The main causes of urban fires are short circuit and gas leakage. Gas leaks in homes generally occur in the kitchen. Then we need a system to be able to read the indication if a fire occurs. By using Smarthome device devices in the home can be controlled and monitored using the help of sensors and microcontrollers. The fire indication detection system is implemented by connecting the microcontroller with the sensor so that the sensor can transmit data. The sensor used in the study is the MQ-135 sensor fire sensor and buzzer. Exchange of data on the system using the MQTT protocol. MQTT is a communication protocol for IoT that applies publish and subscribe. The sensor that serves as a publisher is a fire and smoke sensor while the sensor that serves as a subscriber is a buzzer. The MQTT protocol is the machine to machine communication protocol, can operate at low power and can send messages quickly. Sensors read data from the environment, if the data read exceeds the parameters, it will cause the buzzer to sound. System testing is done by calculating the time of the system in sending data from the sensor to the buzzer and can display data directly and graphically, setting the buzzer and parameters for the sensor. The results of testing the system can read, display data in realtime and can display data in graphical form and speed of sending sensors from the beginning until the buzzer sounds is 1 second.

Keyword: Smarthome, MQTT, mikrokontroler dan Sensor

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

No table of contents entries found.

BAB 1 PENDAHULUAN

1.1 Latar belakang

Kebakaran merupakan sebuah peristiwa dimana ada nyala api baik besar maupun kecil yang bersifat merugikan dan tidak dapat dikendalikan. Api itu sendiri berasal dari reaksi oksidasi dari beberapa zat kimia. Menurut Badan Pencegah Bencana Kota Malang pada tahun 2016 dari bulan januari sampai juni terdapat 10 kejadian kebakaran sedangkan di Kota Jakarta pada tahun yang sama terdapat 1139 kasus kebakaran (Jakarta, 2016). Dapat diambil kesimpulan bahwa semakin padat area maka semakin besar peluang untuk terjadinya kebakaran. Dari banyaknya data kasus terjadi dapat diartikan bahwa upaya pencegahan masih terbilang masih minim. Faktor utama penyebab kebakaran pada area perkotaan adalah arus pendek dan kebocoran gas pada LPG. Untuk itu diperlukan sebuah solusi untuk mendeteksi indikasi adanya kebakaran.

Pada jaman dan era digital seperti ini hampir semua aspek kehidupan tidak pernah lepas dari teknologi. Setiap harinya makin banyak dan makin beragam pula teknologi yang dikembangkan dengan tujuan untuk mempercepat atau mempermudah semua aspek dalam kehidupan manusia. *Internet of Things (IoT)* adalah salah satu konsep yang memanfaatkan keberadaan Internet, dalam *IoT* semua perangkat saling terhubung dan dapat saling berkomunikasi. Salah satu contoh penerapan *IoT* adalah smarthome. Pada smarthome perangkat perangkat dalam rumah seperti tv dan lampu dapat diatur dan dikendalikan sesuai keinginan. Konsep *smarthome* dapat digunakan sebagai salah satu solusi dalam upaya pencegahan terjadinya kebakaran, dengan melakukan *monitoring* atau pengawasan dengan menggunakan perangkat tertentu. Penggunaan mikrokontroler dan sensor banyak digunakan untuk mendukung implementasi *smarthome* (Himanshu Singh, 2018).

Dalam *smarthome* diperlukan suatu protokol untuk melakukan komunikasi antara sensor dan *client*. MQTT Message Queuing Telemetry Transport merupakan protokol komunikasi pada IoT yang berjalan pada TCP/IP yang berbasis *publish* dan *subscribe*. Protokol MQTT digunakan untuk komunikasi mesin ke mesin berdaya rendah serta dapat mengirim dan menerima data dengan ukuran kecil. Pada penelitian yang dilakukan Anusha, Protokol MQTT merupakan protokol yang bisa diimplementasikan pada *smarthome* setelah dibandingkan dengan protokol IoT lainya. Protokol MQTT memiliki kelebihan pada konsumsi daya yang rendah serta dapat mengirimkan paket yang kecil untuk komunikasi mesin ke mesin (Anusha.M, 2017).

Dalam penelitian Akarsh Goyal pada tahun 2016 menyebutkan bahwa MQTT merupakan protokol yang ringan dan cocok untuk peralatan kecil dengan konsumsi daya sedikit serta digunakan dalam lingkungan yang memiliki *bandwith* kecil dan *latency* besar (Akarsh Goyal, 2016). Protokol MQTT dapat diterapkan

pada *smarthome* untuk automatisasi peralatan dalam rumah (Seong-Min Kim, 2015).

Dengan memanfaatkan konsep *IoT* dan menerapkan protokol MQTT dalam mengirimkan data, Oleh karena itu peneliti akan melakukan penelitian tentang "Implementasi Sistem Kebakaran" dengan menggunakan protokol MQTT. Penelitian ini dilakukan menggunakan 2 sensor yaitu sensor MQ-135 untuk asap dan sensor api. Kedua sensor ini nantinya akan menjadi publisher lalu data dari sensor diteruskan menuju mosquitto broker yang nantinya akan diteruskan kepada subscriber dan publisher yaitu aplikasi sistem kemudian data yang didapat nanti akan di update ke database melalui aplikasi dan ditampilkan dalam web jika data dari sensor melebihi parameter, maka aplikasi akan publish kepada sehingga *buzzer* menyala.

1.2 Rumusan masalah

Berdasarkan latar belakang yang ada, maka rumusan masalah yang dapat di buat adalah sebagai berikut:

1. Bagaimana mengimplementasikan protokol MQTT pada sistem pendeteksi kebakaran ?
2. Bagaimana hasil pengujian fungsionalitas pada sistem pendeteksi kebakaran?
3. Bagaimana cara mengetahui kinerja sistem dalam menangani request yang diberikan?

1.3 Tujuan

Berdasarkan latar belakang yang ada, maka rumusan masalah yang dapat di buat adalah sebagai berikut:

1. Dapat melakukan implementasi sistem pendeteksi kebakaran dengan protokol MQTT
2. Untuk mengetahui hasil dari pengujian fungsional pada sistem kebakaran
3. Dapat Membangun sistem sesuai dengan fungsinya serta mengujinya

1.4 Manfaat

Di harapkan denagan menulis penelitian ini penulis dapat menambah ilmu serta memperluas wawasan yang berhubungan dengan komunikasi yang menggunakan protokol MQTT dalam interkasi dengan perangkat lain dan juga hal yang paling penting agar bisa di implemetasikan dalam kehidupan sehai hari serta dapat membantu banyak orang. Untuk Program Studi Teknik Informatika, Jurusan Teknik Informatika serta Fakultas Ilmu Komputer, penelitian ini dapat menambah rujukan penelitian pada bidang jaringan khusunya mengenai konsep Internet Of Things, protokol MQTT, dan sistem indikasi kebakaran.

1.5 Batasan masalah

Untuk pembahsan agar dapat terarahkan pada hal hal yang berkaitan dengan sistem dan tidak menjauh dari latar belakang yang telah di buat maka dapat di jabarkan sebagai berikut:

1. Mikrokontroler yang digunakan dalam penelitian ini adalah Wemos D1 R2
2. Protokol komunikasi yang di gunakan adalah MQTT
4. Database yang di gunakan adalah MySQL
5. Aplikasi yang digunakan oleh pengguna di akses melalui web

1.6 Sistematika pembahasan

BAB I PENDAHULUAN

Berisi tentang latar belakang yang menjelaskan dasar dasar yang mendasari ide dari penelitian ini, rumusan masalah merupakan hal hal yang di anggap oleh penulis perlu untuk di selesaikan, batasan masalah di perlukan agar peneltian tetap tertuju pada ide awal, tujuan bisa dibilang adalah jawaban dari rumusan masalah yang telah di sebutkan, manfaat penelitian, serta sistematika penulisan dari penelitian ini.

BAB II LANDASAN KEPUSTAKAAN

Berisi tentang literatur yang di gunakan dalam penelitian ini mencakup penelitian penelitian yang telah di lakukan sebelumnya yang menunjang penelitian ini serta dasar teori teori yang di gunakan dalam penelitian seperti Mikrokontorller, Protokol MQTT, MySQL, Serta sensor sensor seperti MQ-135, Flame sensor dan Buzzer.

BAB III METODOLOGI PENELITIAN

Berisi tahapan tahapan yang menjelaskan gambaran tentang bagaimana mengimplementasikan penelitian ini seperti mulai membuat ranacangan alur dalam penelitian serta implementasi dan pengujian. Dan juga hal hal seperti perangkat keras, rancangan aplikasi, data yang di proses di bahas secara umum.

BAB IV PERANCANGAN

Berisi tentang desain yang akan di buat agar nanti bisa di implementasikan pada tahap berikutnya yang sesuai deengan penelitian.Hal terkait dengan rancangan sistem alur proses data melalui protokol MQTT, dan juga tahapan tahapan dalam melakukan pengujian.

BAB V IMPLEMENTASI

Merupakan tahapan di mana kita menerapkan perancangan yang sudah di buat sebelumnya sehingga semua sistem dapat terhubung dan dapat berkomunikasi satu sama lain serta tahapan tahpan dalam mengimplentasikan dapat berupa *source code* ataupun hal hal lain yang mendukung dalam implementasi

BAB VI PENGUJIAN DAN ANALISIS HASIL PENGUJIAN

Dalam bab ini merupakan tahapan yang dimana kita menguji sistem yang telah kita buat berdasarkan skenario yang telah di jelaskan pada perancangan hasil dari pengujian nanti kemudian di jabarkan lagi sehingga menjawab tujuan yang di sebutkan pada awal penelitian, dari hasil penelitian tersebut kita dapat mengetahui kinerja dan bagaimana sistem bekerja

BAB VII PENUTUP

Dalam bab ini merupakan thaapan ahir dalam penelitian yang telah di kerjakan dengan memberikan kesimpulan dari semua proses yang telah di lakukan dapat berupa saran ataupun pesan terhadap penelitan yang akan di lakukan selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Dalam bab ini berisi tetang tinjauan pustaka yang meliputi penelitian penelitan yang pernah di lakukan dalam kajian pustaka serta dasar teori yang dibutuhkan dalam penulisan penelitian yang meliputi Smarthome, MQTT, database MySQL, Mikrokontroller Wemos Esp8266 D1 R2 serta berbagai sensor yang ikut terlibat dalam proses penelitian.

2.1 Dasar Teori

2.1.1 Smarthome

Smarthome atau home automation adalah sebuah konsep hunian dimana didalam hunian terdapat sebuah sistem yang berjalan secara otomatis untuk meningkatkan kenyamanan dan keamanan penghuni hunian, smarthome merepresentasikan hunian yang memiliki sistem otomatis yang canggih untuk sistem pencahayaan, sistem keamanan, sistem pengaturan suhu, *home energy management,* sistem multimedia, dan lainnya. Smarthome menghubungkan sistem - sistem otomatis tersebut kedalam satu jaringan yang terintegerasi dengan sebuah pusat kontrol, salah satu tujuan dari smarthome adalah menyesuaikan kebutuhan dari penghuni sesuai dengan standar kenyamanan masing - masing dari penghuni hunian tersebut. Melalui sistem kontrol penghuni dapat mengatur sistem – sistem yang terhubung sesuai dengan kebutuhan penghuni, perangkat - perangkat yang ada di dalam smarthome dapat terhubung dengan menggunakan berbagai cara salah satunya menggunakan tekonologi komunikasi *wireless* seperti *WiFi*. Dalam pengimplementasian smarthome dibutuhkan device – device untuk menunjang seperti perangkat sensor yang akan menjadi pengumpul data dan informasi untuk smarthome, aktuator yang akan menjadi pemicu device – device yang terhubung di dalam sistem, dan aplikasi sistem kontrol yang akan menjadi perantara penghuni dengan sistem smarthome. Aktuator sendiri dapat berupa mikrokontroler, selain itu penggunaan mikrokontroler dapat digunakan untuk kebutuhan pembuatan jaringan di dalam smarthome itu sendiri.

2.2 Kajian Pustaka

Dalam kajian pustaka akan dibahas mengenai bebrapa penelitian penelitian yang pernah dilakukan sebelumnya dan berhubungan serta relevan untuk mendukung penulis terhadap penelitian yang akan di lakukanya.

Penelitian pertama adalah penelitian yang di lakukan (Anusha.M, 2017), penelitian ini membandingkan protokol MQTT, AMQP, CoAP, XMPP, DDS, dan MQTT-SN dari penelitian dari penelitina ini dapat di katakana jika Protokol yang tepat dalam smarthome adalah MQTT.

Penelitian kedua adalah penelitian yang dilakukan oleh (Akarsh Goyal, 2016) pada penelitian ini menggunakan MQTT cloud platform untuk kontroling kondisi pasien secara remote melalui cloud menggunakan platform HiVe MQ, pada penelitian ini menunjukan penggunaan platform MQTT cloud platform dalam hal

ini Hive MQ memiliki waktu transmisi pesan yang kecil, ini dikarenakan kecilnnya paket data yang dikirimkan.

Penelitian ketiga adalah penellitian yang di lakukan oleh (Seong-Min Kim, 2015) Menerapkan proses automatisasi pada smarthome menggunakan protokol MQTT pada perangkat rumah tangga gasr bisa dikontrol dan di monitor secara real time.

Penelitian keempat adalah penelitian yang dilakukan oleh (Sharvari Rautmare, 2016) membandingkan 2 database antara MySQL dan mango db yang dimana dari penelitian tersebut di sebutkan bahwa manggoDB memiliki respon time yang sedikit di banding dengan MySQL serta ManggoDB jauh lebih simple. Dapat di lihat pada Tabel 2.1 merupakan perbandingan dari penelitian sebelumnya dan hubungan dengan penelitian sekarang

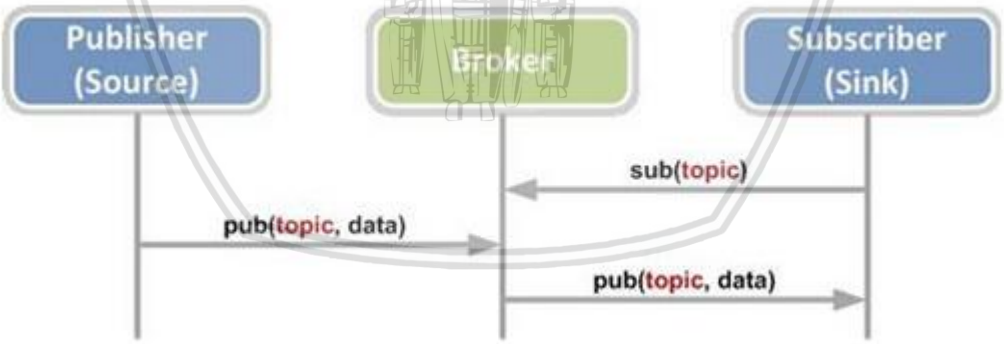
Tabel 2.1 Perbandingan Kajian Pustaka

NO	Judul	Penulis	Perbandingan	
			Kajian Pustaka	Tugas Akhir Penulis
1	Performance Analysis Of Data Protocols of Internet OF Things: A Qualitative Review	Anusha.M, Suresh Babu.E, Sai Mahesh Reddy.L, Vamsi Krishna.A, Bhagyasree.B	Membandingkan Protokol Protokol dalam IoT beserta fungsinya	Menggunakan protokol MQTT dalam komunikasi pertukaran data pada sistem
2	PDConnect: Low Cost Wearable Device connecting Patient-Doctor via Cloud for Good Health	Akarsh Goyal, Ishan Khandelwal, Aditya Kumar Mathur, Ronnie D. Caytiles	Mengimplementasikan Protokol MQTT dalam memntau kondisi pasien	Pengimplentasian protokol komunikasi MQTT agar komunikasi antar perangkat dapat terhubung di dalam sistem
3	IoT Home Gateway for Auto-Configuration and Management of MQTT Devices	Seong-Min Kim, Hoan-Suk Choi, Woo-Seop Rhee	Peroses automatisasi smarthome menggunakan protokol MQTT	Melakukan kontrol dan monitor pada aplikasi

4	MySQL and NoSQL database comparison for IoT application	Sharvari Rautmare	Perbandingan Database MySql dan NoSql untuk IoT	Penggunaan MySQL database untuk apllikasi di karenkan lebih stabil

2.2.1 Message Queue Telemetry Transport (MQTT)

Message Queue Telemetry Transport (MQTT) adalah sebuah protokol komunikasi data machine to machine (M2M) yang berada pada layer aplikasi, MQTT bersifat lightweight message artinya MQTT berkomunikasi dengan mengirimkan data pesan yang memiliki header berukuran kecil yaitu hanya sebesar 2bytes untuk setiap jenis data, sehingga dapat bekerja di dalam lingkungan yang terbatas sumberdayanya seperti kecilnya bandwidth dan terbatasnya sumberdaya listrik, selain itu protokol ini juga menjamin terkiriminya semua pesan walaupun koneksi terputus sementara, protokol MQTT menggunakan metode publish/subscribe untuk metode komunikasinya, cara kerja protokol MQTT dapat dilihat seperti pada gambar **Gambar 2.1** beriikut:



Gambar 2.1 Cara Kerja MQTT

[Sumber tessel.io]

Publish/subscribe sendiri adalah sebuah pola pertukaran pesan di dalam komuunikasi jaringan dimana pengirim data disebut publisher dan penerima data disebut dengan subscriber, metode publish/subscribe memiliki beberapa kelebihan salah satunya yaitu loose coupling atau decouple dimana berarti antara publisher dan subscribe tidak saling mengetahui keberadaannya, terdapat 3 buah decoupling yaitu time decoupling, space decoupling dan synchronization decoupling, time decoupling adalah sebuah kondisi dimana publisher dan

subcriber tidak harus saling aktif pada waktu yang sama, space decoupling adalah dimana publisher dan subscire aktif di waktu yang sama akan tetepi antara publisher dan subcirber tidak saling mengetahui keberadaan dan identitas satu sama lain, dan yang terakhir adalah synchronization decoupling kondisi dimana pengaturan event baik itu penerimaan atau pengiriman pesan di sebuah node hingga tidak saling mengganggu satu sama lain.

Pengiriman data pada MQTT didasari oleh topik, topik ini nantinya yang akan menentukan pesan dari publisher harus dikirim pada subscirber yang mana, topik ini dapat bersifat hirarki, MQTT topic memiliki tipe data string dan untuk perbedaan hirarki atau level dari topik digunakan tanda baca "/", pada MQTT topic dikenal 2 buah wildcard karakter untuk subscription topik yaitu "+" dan "#", penggunaan wildcar karakter "+" adalah sebuah *singel level wildcard* yang digunakan untuk mencockan topik yang disubcriber dengan apapun, contohnya seperti rumah /+/ cahaya, adalah sama dengan rumah/taman/cahaya, atau rumah/kamar/cahaya, tetapi tidak sama dengan rumah/taman, karena karakter "+" adalah single wildcard karakter yang berati mencocokan topik dengan semuanya pada level tertentu, sedangkan karketr "#" adalah *mulit-level wildcard* karakter dimana karakter ini akan mencockan topik pada setiap level karakter ini ada dan setelahnya, contoh rumah/# berarti sama dengan rumah/taman, atau rumah/taman/cahaya atau rumah/cahaya (Solace, n.d.).

Broker merupakan komponen yang paling penting di dalam arsitektur publish/subcsribe dan MQTT, broker adalah sebagai perantara pertukaran pesan antara publisher dan subscriber, masing - masing dari publisher dan subscriber harus terkoneksi ke broker untuk mengirimkan ataupun menerima pesan, broker akan menerima seluruh pesan yang di publish oleh publisher, untuk selanjutnya diteruskan kepada subscriber berdasarkan dengan topik yang sesuai dengan pesan yang dikirim dan topik yang disubscribe oleh subscriber, ketika subscriber tidak aktif maka broker akan menyimpan pesan pada buffer untuk selanjutnya dikirimkan ketika subscriber telah aktif kembali.

MQTT memiliki 3 level quality of service (QOS) dalam pengiriman pesannya yaitu 0,1,2, Pada level 0 atau disebut dengan "*at most once delivery message"* QoS level 0 ini adalah QoS paling cepat untuk mengirim pesan, QoS ini akan menjamin mencoba mengirim dengan usaha terbaiknya,pesan akan langsung dikirimkan tanpa menunggu acknowledge dari sisi penerima, namun pengiriman dengan menggunakan QoS level 0 memungkinkan hilangnya pesan jika penerima terputus secara tiba – tiba, QoS level 1 atau disebut juga "*at least once*" dengan opsi level ini MQTT akan mengirimkan pesan minimal sekali, pesan yang dikirimkan nantinya akan di acknowledge oleh penerima, MQTT akan mengirimkan kembali pesan yang menggunakan QoS level 1 jika pengirim tidak menerima acknowledge dari penerima, pesan yang dikirimkakn menggunakan QoS level 1 juga akan di simpan oleh broker pada database, untuk QoS level 2 atau "exactly once delivery" QoS ini akan menjami pesan dikirim dan diterima pada sisi penerima, QoS ini adalah yang paling aman namun juga paling lambat dibandingkan dengan 2 level QoS lainnya,

jaminan pesan akan disampai dikarenakan adanya komunikasi 2 arah dari pengirim dan penerima (Lampkin, et al., 2012).

2.2.1.1 Mosquitto Broker

Mosquitto broker adalah salah satu opensource broker pesan yang mengimplementasikan protokol MQTT versi 3.1 dan 3.1.1, broker mosquitto juga mendukung implementasi server *lightweight* dari MQTT maupun MQTT-SN, mosquitto broker ditulis dalam bahasa pemrogram C dengan alasan agar dapat tetap bekerja pada mesin yang tidak mendukung JVM, dari hasil pengujian yang telah dilakukan sebelumnya broker mosquitto dapat mendukung 100.000 koneksi secara bersamaan berfungsi sebagai penghubung antara *publisher* dan *subscriber* (Eclipse, 2013).

2.2.1.2 Paho MQTT

Paho MQTT adalah sebuah library untuk MQTT client yang dikembangkan oleh eclipse, paho library tersedia untuk berbagai bahasa pemrogran seperti Lua, Python, C++ dan Javascript. Paho MQTT mendukung penggunakan 3 level qos MQTT untuk pengriman/publish pesan dan juga untuk penerimaan/subscribe pesan (Eclipse, 2011).

2.2.2 Mikrokontroler

Mikrokontroler adalah sebuah papan rangkaian elektronik yang didalamnya sudah terdapat *chip* mikrokontroler atau cpu kecil, memory dan interface Input/Output, mikrokontroller banyak digunakan untuk embeded sistem dan komunikasi *machine to machine* (M2M) memiliki beberapa keunggulan seperti hemat sumberdaya, fleksibel dan harganya yang murah.

2.2.2.1 *Wemos D1 R2*

Wemos D1 R2 adalah sebuah mikrokontroler yang kompetibel/mirip dengan arduino uno hanya saja wemos D1 R2 berbasis modul ESP8266-12, bahasa pemrograman yang digunakan untuk memprogram wemos D1R2 ini adalah bahasa pemrograman C namun modul esp8266 sudah memiliki cukup banyak library untuk digunakan sehingga pemrograman mikrokontroler berbasis modul esp8266 menjadi relatif mudah meskipun untuk pemula, untuk melakukan pemrograman pada board Wemos D1 R2 ini dapat menggunakan aplikasi Arduino IDE, wemos D1 R2 memiliki 11 digitan input/output pins, 1 analog input pin, microusb untuk koneksi, dan power jack 9-24V daya input (Wemos, n.d.), bentuk fisik Wemos D1 R2 ditunjukan pada **Gambar 2.2** berikut:

Gambar 2.2 Bentuk Fisik Wemos D1 R2

[sumber : instructables.com]

Wemos D1 R2 dengan modul esp-8266 memiliki spesifikasi teknis seperti yang diterangkan pada **Tabel 2.2** berikut:

Microkontroler	**ESP-8266**
Daya Operasi	3.3V
Digital I/O pins	11
Analog Input pins	1(max input 3.2V)
Kecepatan Clock	80MHz/160MHz
Flash	4M bytes
Panjang	68.6mm
Lebar	53.4 mm
Berat	25g

Tabel 2.2 Spesifikasi Teknis Wemos D1 R2 Esp8266

Semua pin pada wemos D1 R2 bekerja pada daya 3.3V untuk spesifikasi dari pin yang dimiliki oleh Wemos D1 R2 ini diterangkan pada **Tabel 2.3** berikut:

PIN	**Function**	**ESP-8266 Pin**
TX	TXD	TXD
RX	RXD	RXD
A0	Analog Input	A0
D0	IO	GPIO16
D1	IO,SCL	GPIO5
D2	IO,SDA	GPIO4
D3	IO,10K pull-up	GP1O0
D4	IO,10K pull-up,Builtin_LED	GPIO2

D5	IO, SCK	GPIO14
D6	IO,MISO	GPIO12
D7	IO,MOSI	GPIO13
D8	IO, 10k Pulldown SS	GPIO15
G	Ground	GND
5V	5V	5V
3V3	3.3V	3.3V
RST	Reset	RST

Tabel 2.3 Tabel Pins Wemos D1 R2 Esp8266

2.2.2.2 *Esp8266*

Esp8266 adalah modul wifi yang terintegerasi dengan protokol TCP/IP, dan memiliki kapabilitas sebagai *micro processor unit* (MCU), yang berarti esp8266 ini memberikan kemampuan kepada semua mikrokontroler untuk mengakses jaringan WIFI yang tersedia. Modul Esp8266 cukup kuat untuk melakukan proses dan kapabilitas penyimpanan datanya, yang memungkinkan modul ini untuk terintegerasi dengan sensor ataupun aplikasi lain yang spesifik melalu GPIOs miliknya, versi terbaru dari ESP8266 wifi modul memiliki peningkatan ukuran flash disk dari 512KB menjadi 1MB. Esp8266 memiliki fitur sebagai berikut (sparkfun, n.d.):

- 802.11 b/g/n
- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLLs, regulators, DCXO and power management units
- +19.5dBm output power in 802.11b mode
- Power down leakage current of <10uA
- 1MB Flash Memory
- Integrated low power 32-bit CPU could be used as application processor
- SDIO 1.1 / 2.0, SPI, UART
- STBC, 1×1 MIMO, 2×1 MIMO
- A-MPDU & A-MSDU aggregation & 0.4ms guard interval

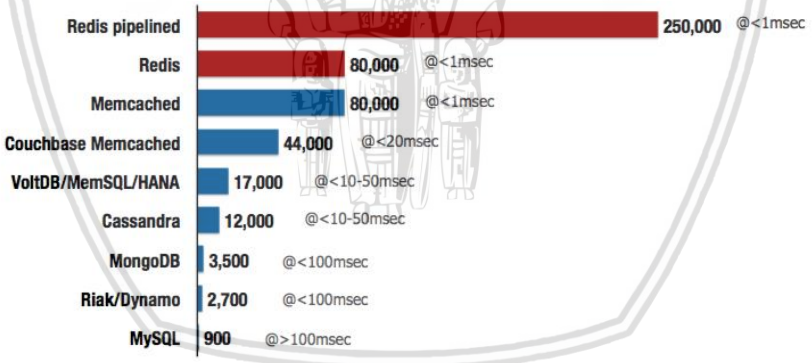
- Wake up and transmit packets in < 2ms
- Standby power consumption of < 1.0mW (DTIM3)

2.2.3 Database

2.2.3.1 *MySQL*

Remote Dictioary Server (MySQL) adalah database tipe database *nosql* yang bersifat *key-value* dan penyimpanan datanya di dalam memori, MySQL biasanya digunakan sebagai database, cache, maupun broker MySQL mendukung berbagai jenis data struktur seperti string, hashes, lists, sets, sorted sets dengan query, bitmaps, hyperloglogs, dan geospatial index.

Untuk dapat meraih performa yang bagus MySQL bekerja dengan memory dataset, bergantung dengan berbagai kasus penggunaan MySQL juga dapat membackup datanya pada disk sesuai dengan pengaturan yang dilakukan, MySQL sendi memiliki konsep yang mirip dengan database – database lainnya akan tetapi berbeda seperti database yang lain, yang pada umumnya data ditampilkan pada sebuah aplikasi secara bersamaan, MySQL memerlukan perintah pada *command line interface* (CLI) nya untuk menampilkan datanya, MySQL sendiri mendukung fitur expire sehingga dalam penggunaannya tidak perlu khawatir untuk kehabisan memori, kecepatan performa dari MySQL dapat di lihat pada **Gambar 2.3** berikut:



Gambar 2.3 Perbandingan Performa MySQL dengan Database Lain

[sumber : MySQLlabs.com]

2.2.4 Sensor

2.2.4.1 *MQ-135*

MQ-135 Air Quality Sensor adalah sensor yang memonitor kualitas udara untuk mendeteksi gas amonia (NH3), natrium-(di)oksida (NOx), alkohol / ethanol (C2H5OH), benzena (C6H6), karbondioksida (CO2), gas belerang / sulfur-hidroksida (H2S) dan asap / gas-gas lainnya di udara.

Sensor ini melaporkan hasil deteksi kualitas udara berupa perubahan nilai resistensi analog di pin keluarannya. Pin keluaran ini bisa disambungkan dengan pin ADC (analog-to-digital converter) di mikrokontroler / pin analog input Arduino.

Gambar 2.3 Bentuk FisikMQ-135

[Sumber : http://www.tokobagus.com]

2.2.4.2 *Sensor Api*

Flame detector merupakan alat optik yang digunakan untuk mendeteksi nyala api. Prinsip kerja dari alat ini adalah mendeteksi radiasi infra-red atau ultraviolet dari api yang menyala. Flame Detektor umumnya akan merespon jauh lebih cepat misalnya terjadi kebakaran yang diakibatkan oleh gas dan cairan yang mudah dibakar. Namun, flame detector tidak efektif digunakan jika kebakaran yang terjadi lambat. Flame detector memiliki banyak ukuran dan variasi namun secara umum yang perlu diketahui hanya terbagi menjadi tiga kelompok antara lain infra red flame detector dan ultraviolet flame detector serta gabungan antara keduanya.

Gambar 2.4 Bentuk Fisik Arduino Compatible IR

[Sumber : http://www.bukalapak.com]

2.2.4.3 *Buzzer*

Buzzer adalah sebuah komponen elektronika yang berfungsi untuk mengubah getaran listrik menjadi getaran suara. Pada dasarnya prinsip kerja buzzer hampir sama dengan loud speaker, jadi buzzer juga terdiri dari kumparan yang terpasang pada diafragma dan kemudian kumparan tersebut dialiri arus sehingga menjadi elektromagnet, kumparan tadi akan tertarik ke dalam atau keluar, tergantung dari arah arus dan polaritas magnetnya, karena kumparan dipasang pada diafragma maka setiap gerakan kumparan akan menggerakkan diafragma secara bolak-balik sehingga membuat udara bergetar yang akan menghasilkan suara. Buzzer biasa digunakan sebagai indikator bahwa proses telah selesai atau terjadi suatu kesalahan pada sebuah alat (alarm). **Gambar 2.5**

Gambar 2.5 Bentuk Buzzer

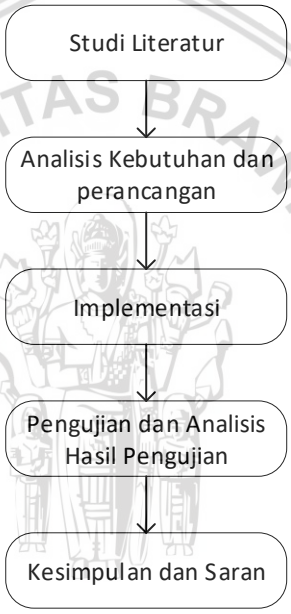
BAB 3 METODOLOGI

3.1 Jenis Penelitian

Jenis penelitian yang dilakukan oleh penulis adalah penelitian implementatif dan perancangan (*design*). Penelitian jenis ini dalam rangka membuat sebuah perangkat lunak dan sebuah sistem, penelitian dimulai dengan analisa kebutuhan, pembuatan perangkat lunak dan sistem, serta pengujian produk perangkat lunak dan sistem.

3.2 Metodologi Penelitian

Alur dari Metodologi yang akan dilakukan adalah sebagai berikut di tunjukan oleh Gambar 3.1



Gambar 3.1 Alur Metode Penelitian

3.2.1 Studi Literatur

Studi literatur merupakan tahap dimana belajar dan lebih mematangkan konsep dari semua jenis informasi yang bisa membantu untuk proses pengerjaan penelitian, untuk penelitian kali ini jenis literatur yang paling sering dipakai adalah jurnal. Kategori untuk studi literatur itu sendiri dapat diringkas menjadi seperti berikut:

1.MQTT

2.Smarthome

3.Mikrokontroler

4.MySQL

5.Sensor MQ-135

6.Sensor Api

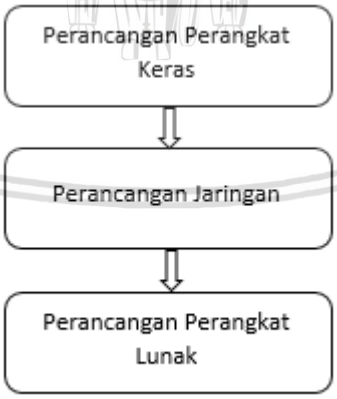
7.Buzzer

3.2.2 Analisis Kebutuhan

Pada tahapan saat ini diharuskan untuk mengetahui kebutuhan baik dari segi perangkat keras maupun perangkat lunak. Untuk itu diperlukan analisis yang baik untuk mempermudah pengerjaan penlitian pada tahapan selanjutnya agar sumber daya yang akan kita gunakan tidak akan terbuang percuma ataupun kurang. Untuk merancang sistem diperlukan juga analisis kebutuhan sehingga menjadi tepat sasaran dan terarah serta memperkecil kesalahan dalam implementasi dari segi perangkat keras maupun dari segi perangkat lunak. Salah satu cara yang digunakan untuk menganalisis kebutuhan adalah dengan cara mempelajari literatur tentang penelitian yang sejenis yang sudah pernah dilakukan untuk mengetahui apa saja yang dibutuhkan dan menganalisis kekurangan dari penelitian yang sudah ada.

3.2.3 Perancangan

Secara umum perancangan pada penelitian ini dibagi menjadi 3 yaitu perancangan perangkat keras, perancangan perangkat lunak, dan perancangan jaringan. Tujuan dari perancangan itu sendiri untuk memudahkan peneliti dalam hal fisik dan mempunyai gambaran seperti apa nanti implementasi pada perangkat yang akan digunakan. Alur yang dilakukan dapat dilihat pada Gambar 3.2**:**



Gambar 3.2 Perancangan Jaringan

Untuk perancangan yang pertama adalah perangkat keras pada tahap ini peneliti harus mampu menghubungkan semua perangkat keras agar dapat berkomunikasi satu sama lain mampu berkomuniasi antar mikrokontroler, mampu berkomunikasi dengan sensor yang terdiri dari sendor MQ-135, sensor api, dan mampu berkomunikasi dengan *buzzer*.

Untuk perancangan kedua yaitu perancangan jaringan diharapkan mampu menjelasakan alur dimana mikrokontroler yang mendapat data dari sensor dapat mengirimkan data menuju broker, dari broker menuju sistem perangkat lunak menggunakan protokol MQTT kemudian dari sistem perangkat lunak diteruskan melalui broker menuju buzzer menggunakan metode *publish dan subscribe*.

Untuk perancangan yang ketiga yaitu perancangan perangkat lunak yang berfokus pada aplikasi yang dibuat beserta fitur yang dapat dilakukan seperti memonitor keadaan sensor dan juga mengkontrol parameter serta inputan untuk mengatur *buzzer*, semua yang dikontrol dan dimonitor nanti akan tampil pada layar dimana aplikasi berjalan. Perancangan perangkat lunak hampir meliputi semua mulai dari hasil sensor yang berupa asap, api lalu mengkontrol *buzzer* dan database MySQL.

3.2.4 Implementasi

Pada perancangan telah terjabarkan bagaimana nantinya implementasi secara keseluruhan. Pada umumnya implementasi dibagi menjadi 3 yaitu:

1. Implementasi Perangkat keras yang bertujuan agar semua perangkat dapat berkomunikasi dan saling terhubung
    - Konfigurasi mikrokontroler Wemos Esp8266 D1 R2 dengan sensor MQ-135
    - Konfigurasi mikrokontroler Wemos Esp8266 D1 R2 dengan sensor api
    - Konfigurasi mikrokontroler Wemos Esp8266 D1 R2 dengan *buzzer*
2. Implementasi Jaringan semua perangkat harus terhubung dengan akses point yang ada pada salah satu mikrokontroler, terjadi pembagian tugas antar mikrokontroler sebagi *publisher* atau *subscriber* serta dapat berkomunikasi dengan broker yaitu Mosquitto serta *publish* atau *subsribe* pada aplikasi perangkat lunak
3. Implementasi perangkat lunak memiliki beberapa hal yang harus dikerjakan semuanya dapat dibagi menjadi:
    - Instalasi mosquitto broker, MySQL dan arduino ide sebagai dasar dari perancangan dan implementasi dalam penelitian ini
    - Aplikasi, perangkat lunak yang merupakan *interface* bagi pengguna untuk mengakses, menggunakan serta melakukan kendali dan kontroling terhadap sistem
    - Database, merupakan tempat untuk menyimpan data dari sensor pada dan untuk menampilkan pada aplikasi perangkat lunak secara *localhost* yang berada dalam komputer menggunakan MySQL sebagai database

- Monitoring, proses dimana dapat melihat *output* dari sensor berupa grafik dan juga nilai dari sensor secara *realtime* memudahkan pengguna dalam membaca dan menggunakan aplikasi perangkat lunak
- Kontroling, kondisi dimana dapat merubah atau mengatur inputan sehingga dapat melakukan aksi seperti yang diminta oleh user

3.2.5 Pengujian dan Analisa Hasil Pengujian

Penujian dibagi menjadai 2 jenis yaitu pengujian aplikasi perangkat lunak dan pengujian serta analisa hasil untuk pengujian pertama berfokus pada kinerja aplikasi perangkat lunak serta mengukur keberhasilan dan kekurangan terihat dari bagaimana pengiriman data dan hasil yang dikeluarkan adapun beberapa sekenario yang akan di lakukan adalah:

1. Memastikan bahwa semua perangkat telah terhubung
2. Berhasil menerapkan protokol MQTT serta implementasi *publish* dan *subscribe*
3. Memastikan bahwa perangkat dapat mengirimkan data dan menerima data melalui broker

Untuk pengujian dan analisa dari hasil aplikasi perangkat bertujuan untuk mengtahui jika aplikasi perangkat lunak yang telah dibuat sudah sesuai dan memenuhi tujuan awal yang telah ditetapkan. Terdapat beberapa pengujian untuk mengukur hal tersebut yaitu dengan menguji seberapa cepat sistem berinteraksi dengan sensor.

BAB 4 ANALISA KEBUTUHAN DAN PERANCANGAN

Untuk mendapatkan aplikasi perangkat lunak yang baik dibutuhkan analisa terhadap perangkat keras maupun perangkat lunak. Perancangan adalah tahapan sebelum melakukan implementasi sehingga perancangan yang baik dapat membantu pada tahapan selanjutnya, ada beberapa hal yang akan dilakukan pada perancangan ini yaitu perancangan perangkat keras serta perancangan perangkat lunak.

4.1 Analisis Kebutuhan

Pada penjelasan singkat kalimat diatas untuk analisa kebutuhan akan dijelaskan tentang perangkat perangkat yang akan digunakan secara lebih rinci baik dari perangkat keras maupun perangkat lunak untuk lebihnya analisis di jabarkan sebagai berikut:

1.Perangkat keras

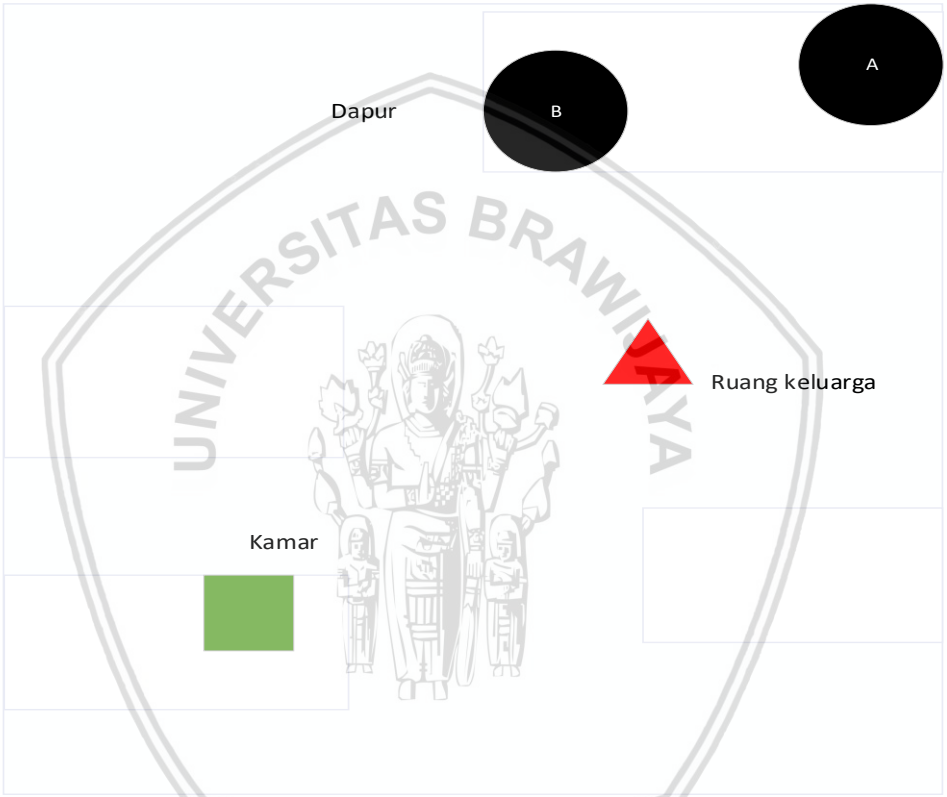
- Mikrokontroler Wemos D1 R2, perangkat ini digunakan sebagai media untuk sensor agar bisa membaca data serta mengirimkan data tersebut dan juga sebagai penghubung antar melalui Esp822 yang sudah ada didalamnya
- Sensor MQ-135, sensor ini berperan untuk membaca jika ada asap dan mengirim data
- Sensor api, sensor ini berperan untuk membaca jika terlihat api dan mengirim data
- *Buzzer*, merupakan sensor yang berfungsi sebagai pemberi tahu jika ada api atau asap
- Komputer, merupakan tempat dimana broker dan aplikasi perangkat lunak berada

2.Perangkat lunak

- Program Arduino IDE untuk mikrokontroler, digunakan untuk mengatur sensor sehingga bisa terkoneksi dengan broker serta dapat membaca dan mengirimkan data untuk bahasa program ini menggukan bahasa C
- Mosquitto broker, merupakan penghubung dalam protokol MQTT yang dimana nanti semua pertukaran data akan melaui broker
- Database MySQL tempat penyimpanan semua data yang telah dikirimkan oleh broker
- Program aplikasi perangkat lunak yang ditulis menggukan bahasa pemograman Python menjadi penghubung antara pengguna dengan sistem di harapkan mampu menjalankan fungsinya dalam hal monitoring dan kontroling

4.2 Perancangan

Pada penilitan ini akan di lakukan sebuah percobaan yang sederhana dimana nanti sensor MQ-135 yang bertugas untuk mendeteksi asap kemudian sensor api untuk mendeteksi adanya api. Data dari kedua sensor tersebut dikirm menuju broker jika data melebihi parameter yang di tentukan maka akan membuat *buzzer* berbunyi yang dikontrol dan dimonitor oleh aplikasi perangkat lunak serta dapat mematikan *buzzer* atau mengatur secara otomatis untuk gambaran tempat dan bagaimana sensor diletakan bisa dilihat pada Gambar 4.1:



Gambar 4.1 Topologi Perancangan

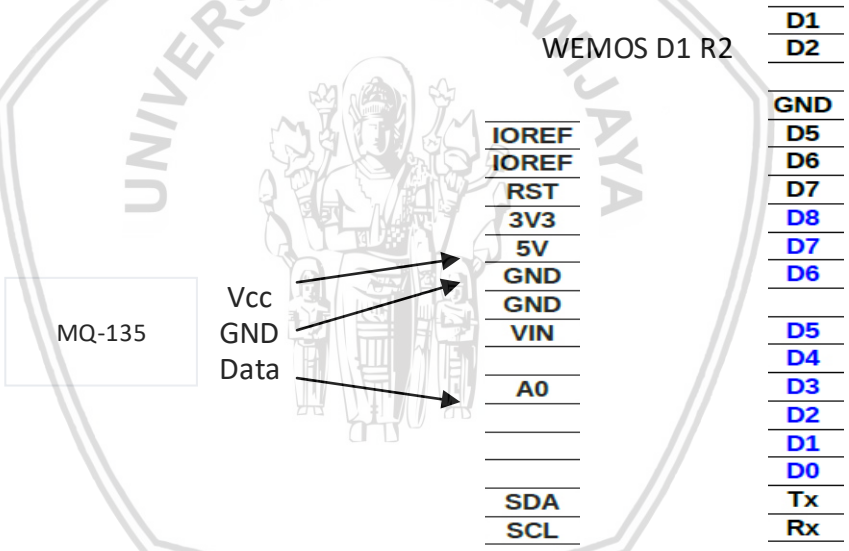
Untuk lingkaran adalah mikrokontroler yang terhubung dengan sensor, untuk segitiga adalah mikrokontroler yang terhubung dengan *buzzer*, untuk persegi adalah laptop yang sudah terpasang broker serta aplikasi perangkat lunak yang nantinya diharapkan jika terjadi sesuatu pada daerah dapur dari kebakaran, indikasi kebocoran gas, dan asap maka dapat secara langsung terkontrol dan termonitor dan dapat dilakukan tindakan pencegahan terlebih dahulu.

4.2.1 Perancangan Perangkat Keras

Untuk menghubungkan sensor dengan mikrokontroler dibutuhkan perancangan untuk menghindari kesalahan yang bisa menyebabkan kerusakan pada perangkat ataupun sensor. Menghubungkan perangkat ini bertujuan agar sensor dapat mengirimkan data sesuai dengan yang diperintahkan melalui mikrokontroler adapun perangkat yang akan dihubungkan adalah mikrokontroler dengan sensor MQ-135, sensor api dan *buzzer*.

4.2.1.1 Perancangan Sensor MQ-135 dengan Mikrokontroler Wemos D1 R2

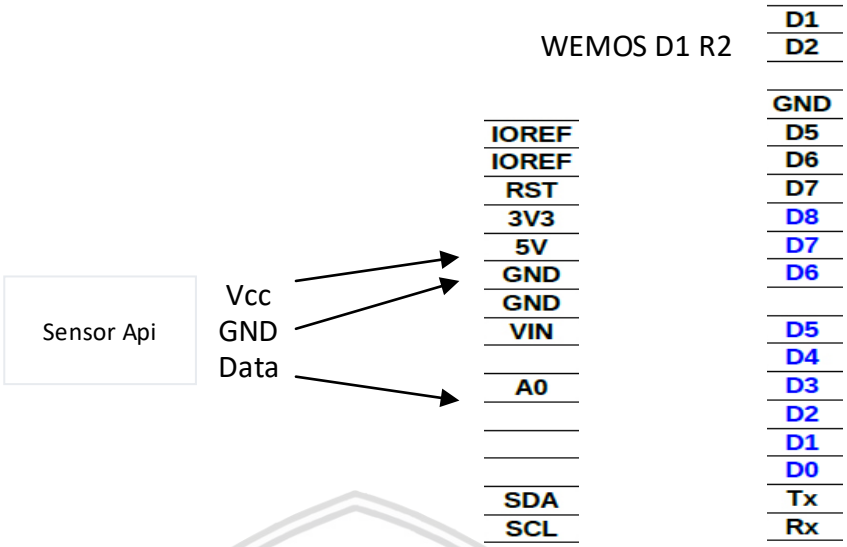
Sensor MQ – 135 mempunyai 4 pin yaitu Vcc, GND, analog *output*, digital *output*, Vcc nantinya akan disambungkan dengan sumber daya dari mikrokontroler lalu GND juga dihubungkan pada salah satu pin mikrokontroler. Karena data berjenis analog akan dipasangkan pada pin analog yang tersedia, sedangkan pin digital tidak terpakai untuk lebih rincinya dapat dilihat pada Gambar 4.2:



Gambar 4.2 Perancangan Mikrokontroler Wemos D1 R2 dengan Sensor MQ-135

4.2.1.2 Perancangan Sensor Api dengan Mikrokontroler Wemos D1 R2

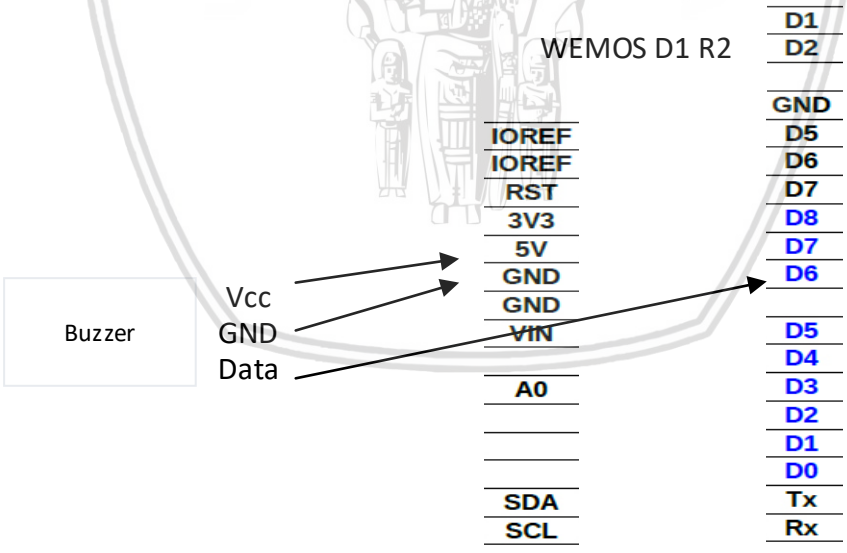
Sensor api juga mempunyai jumlah pin yang sama yaitu 4 VCC, GND, analog *output*, Vcc juga nanti akan dihubungkan pada sumber daya digital dan GND dihubungankan dengan Ground yang ada pada mikrokontroler dan analog *output* juga dihubungan dengan pin analog yang ada pada mikrokontroler sedangkan digital output tidak digunakan untuk lebih rincinya dapat dilihat pada Gambar 4.3:



Gambar 4.3 Perancangan Mikrokontroler Wemos D1 R2 dengan Sensor Api

4.2.1.3 Perancangan *Buzzer* dengan Mikrokontroler Wemos D1 R2

Sensor *buzzer* mempunyai 3 pin yaitu VCC, GND, I/O, untuk VCC nanti akan dihubungkan dengan sumber daya yang ada pada mikrokontroler, untuk GND akan disambungkan dengan pin ground yang ada pada mikrokontroler juga dan karena I/O merupakan pin digital maka akan disambungkan dengan pin digital yang ada untuk lebih rincinya dapat dilihat pada Gambar 4.4:



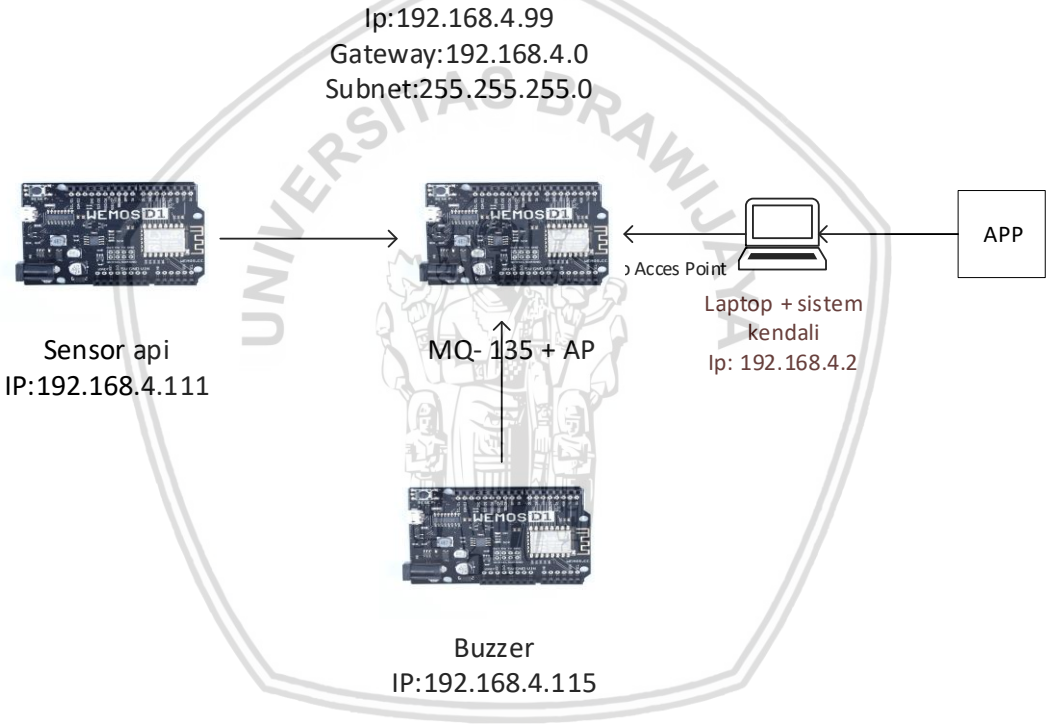
Gambar 4.4 Perancangan *Buzzer* dengan Mikrokontroler Wemos D1 R2

4.2.2 Perancangan Jaringan

Perancangan jaringan ini ditujukan agar perangkat dapat terhubung dan berinterkasi dengan cara memberi alamat pada tiap perangkat serta alur alur yang terjadi sesuai dengan tugas.

4.2.2.1 Perancangan Arsitektur Jaringan

Pada tahapan ini kita memberi alamat kepada semua perangkat sehingga bisa membedakan perangkat satu dengan yang lain dan juga untuk membagi tugas agar tidak bertabrakan jika mempunyai alamat yg sama nantinya semua perangkat akan terhubung pada salah satu mikrokontroler yang menjadi AP dengan menggunakan modul Esp8266 yang memiliki fitur baik menjadi AP maupun hanya menjadi client saja. Untuk lebih detail pengalamatan perangkat dapat di lihat pada Gambar 4.5:



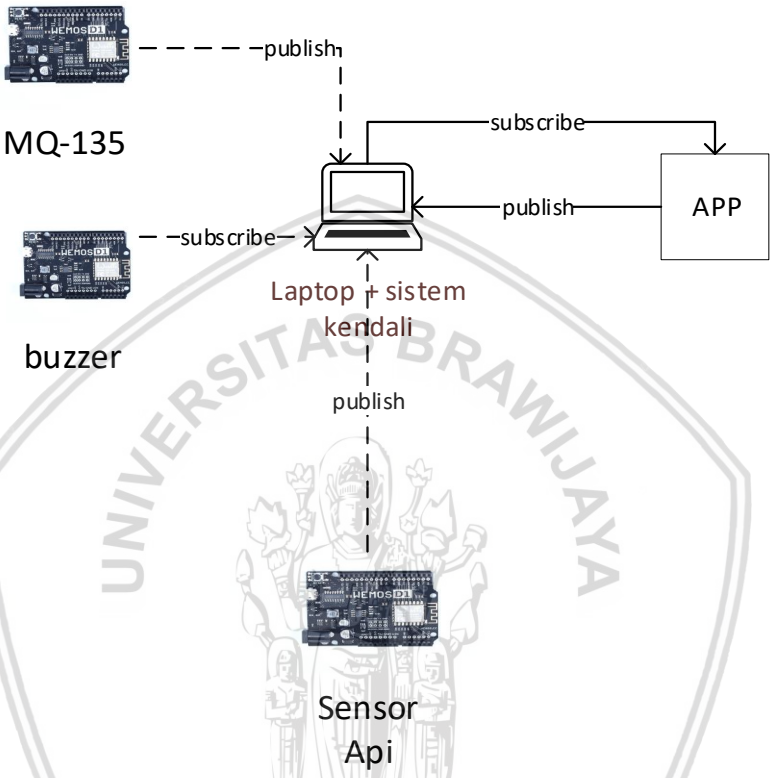
Gambar 4.5 Rancangan Alur Pengalamatan dan Arsitetur Jaringan

4.2.2.2 Perancangan Integrasi Perangkat Dengan Mosquitto Broker

Sebelum data dapat diterima dan ditampilkan terlebih dulu semua perangkat harus tersambung dengan broker karena pada MQTT semua pertukaran data yg terjadi harus melalui broker terlebih dahulu, broker terletak pada komputer sama dengan aplikasi perangkat lunak semua mikrokontroler yang telah terintegrasi dengan sensor harus bisa berkomunikasi dengan broker karena data tidak akan sampai ke sistem jika tidak melewati broker.

4.2.2.3 Perancangan *Publish* dan *Subcribe*

Setelah terkoneksi oleh broker lalu kita membagi tugas siapa yang menjadi *publisher* dan siapa yang menjadi *subscriber* untuk *publisher* harus menuliskan topik saat mau mengirim data begitu pula dengan *subscriber* alur *publish* dan *subscribe* dapat di lihat lebih detail pada Gambar 4.6



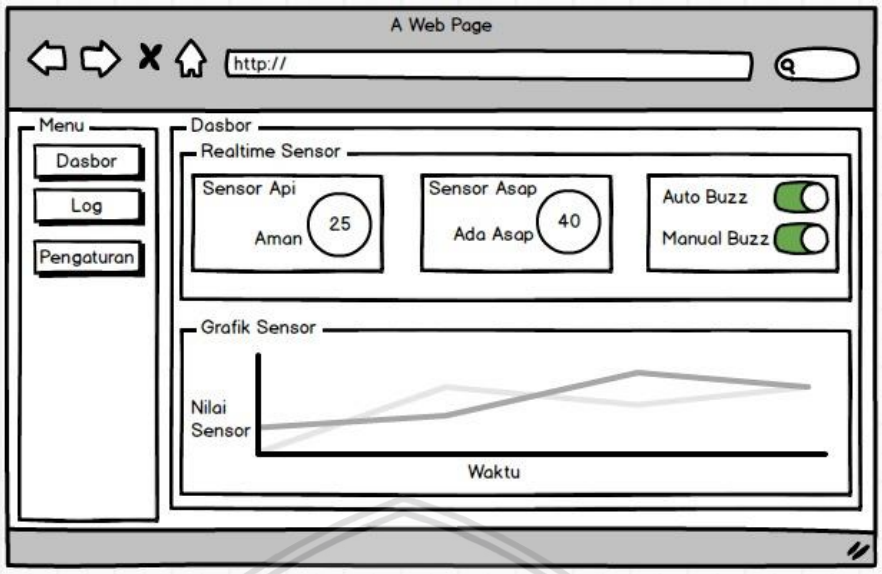
Gambar 4.6 Alur *Publish* dan *Subscribe* Pada Sistem

4.2.3 Perancangan Perangkat Lunak

Perancangan yang didesain untuk mengatur semua yang berhubungan dengan perangkat lunak yang terdiri dan tugas yang dilakukan oleh perangkat lunak tersebut.

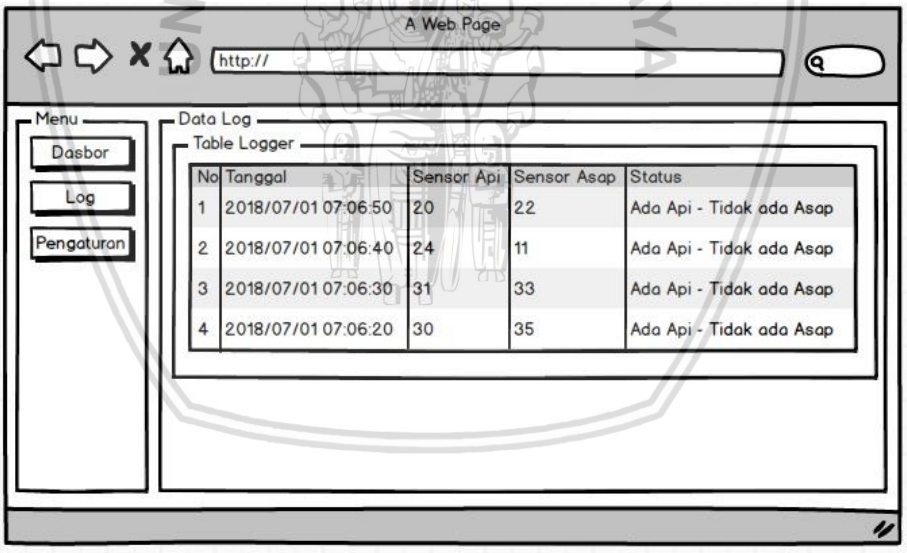
4.2.3.1 Perancangan Aplikasi Perangkat Lunak

Dalam aplikasi perangkat lunak yang dibuat ini harus bisa menjalankan tugas untuk memonitor dan mengkontrol data data dari perangkat itu sendiri serta mengontrol perangkat yang sudah saling terhubung. Ada 3 menu dalam rancangan ini untuk menu yang pertama dapat dilihat pada Gambar 4.7:



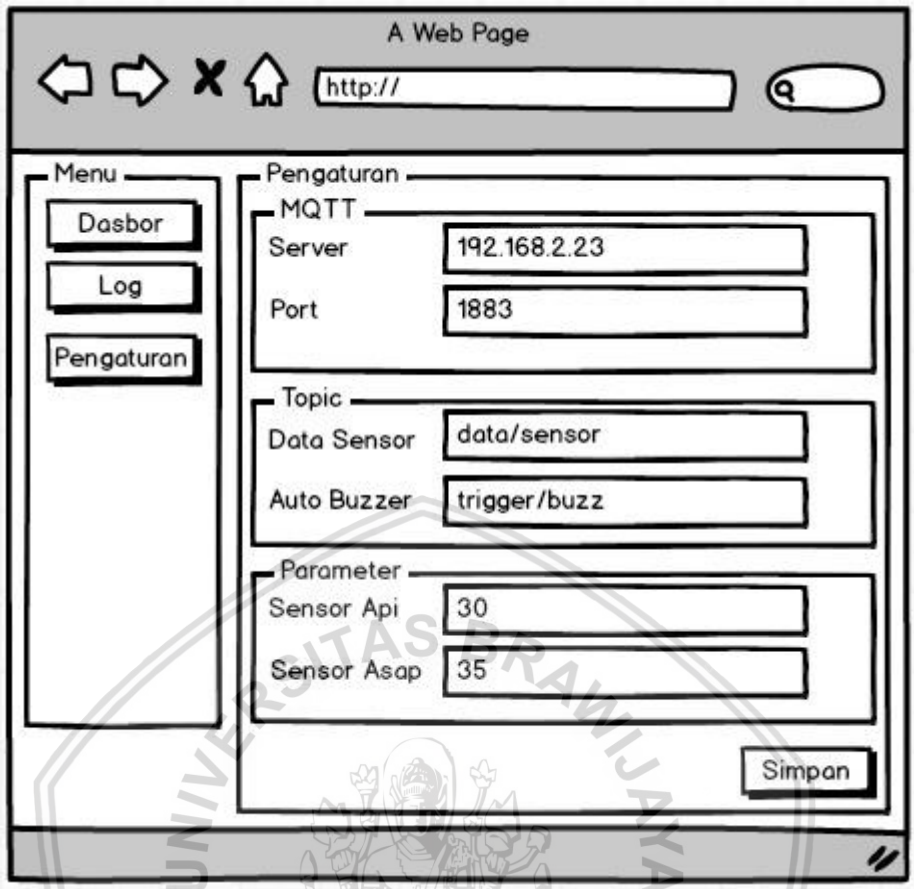
Gambar 4.7 Tampilan Aplikasi Perangkat Lunak Dasbor

Pada menu dasbor nantinya didapatkan data dari kedua sensor serta dapat terlihat juga status sensornya dan dapat mengkontrol *buzzer*. Dapat dilihat dari grafik sensor disana nantinya data yang kita dapat akan disuguhkan dalam bentuk yg lebih interaktif. Untuk Menu Log dapat kita lihat pada Gambar 4.8**:**



Gambar 4.8 Tampilan Aplikasi Perangkat Lunak Log

Pada menu ini befungsi agar pengguna dapat melihat catatan data yang ditangkap oleh dan disimpan sebgai catatan. Menu terahir dari aplikasi perangkat lunak adalah pengaturan yang dapat kita lihat pada Gambar 4.9:



Gambar 4.9 Tampilan Aplikasi Perangkat Lunak Pengaturan

Pada menu ini merupakan menu dimana kita bisa mengkontrol inputan untuk Broker, Topik dan parameter.

4.2.4 Perancangan Database

Databse yang digunakan dalam penelitian ini adalah MySQL yang nantinya akan menyimpan data dari hasil *output* sensor yg terlebih dahulu berada pada aplikasi perangkat lunak untuk proses penyimpanan data menggunakan satu tabel seperti pada dengan 1 merupakan data dari asap dan 0 dari api Gambar 4.1**:**

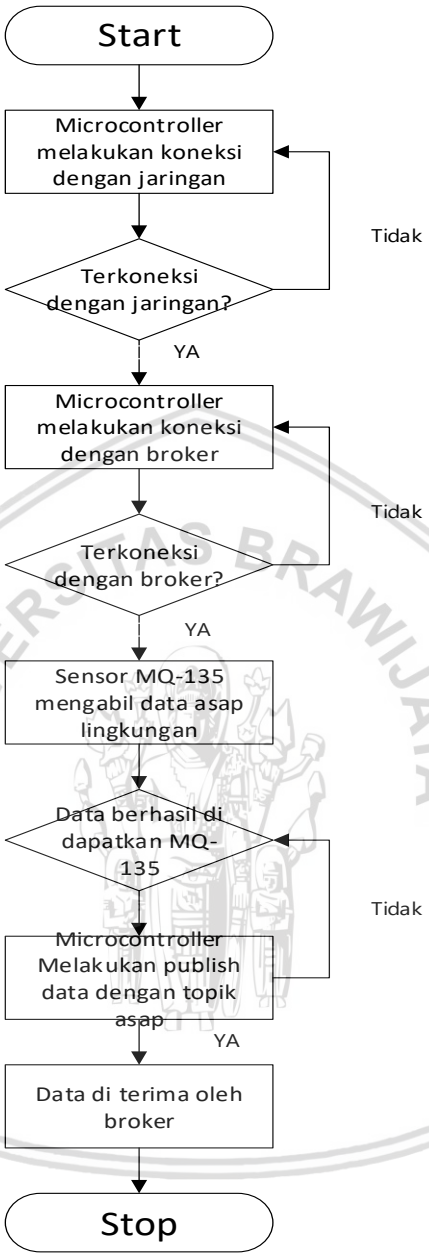
Tabel 4.1 Rancangan Database pada MySQL

Sensor	Hasil
Asap (1)	10
Api (0)	20

4.2.5 Perancangan Monitoring Sensor Api dan Sensor Asap

Untuk mengetahui alur dan tahapan bagaimana data dari sensor didapatkan lalau diproeses serta ditampilkan dalam aplikasi maka diperlukan perancangan

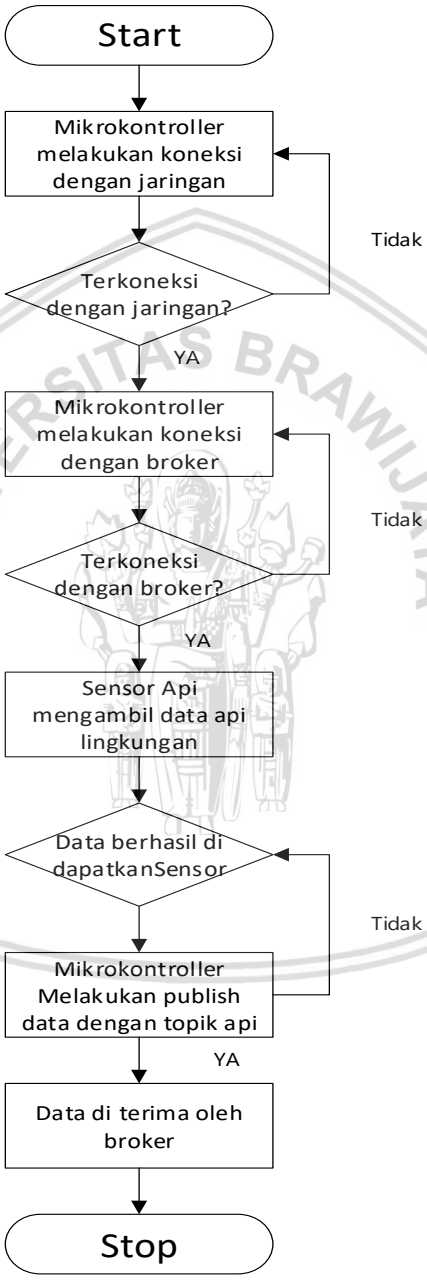
sehingga alur dan tahapan dapat dibaca dan dipahami pada Gambar 4.10 dapat dilihat proses dan alur bagaimana sensor mengirim data:



Gambar 4.10 Proses Pengiriman Data pada Sensor MQ-135

Sebelum data dapat terkirim diharuskan pada semua perangkat yaitu sensor dan mikrokontroler harus terhubung telebih dahulu, setelah itu baru sensor dapat membaca keadaan lingkungan dengan cara membakar kertas dan hasil pembakaran tersebut dapat menghasilkan asap lalu data yang dibaca tersebut kemudian dikirimkan menuju broker menggunakan protokol MQTT, Jika kita membicarakan MQTT maka pasti ada yang dinamakan *publish* dan *subscribe.* Untuk proses pengiriman data ini disebut *publish* data yang telah dikirim oleh sensor melalui mikrokontroler menggunakan format (topik,nilai) karena data sensor tidak bisa terbaca secara langsung dan juga pengiriman dalam protokol

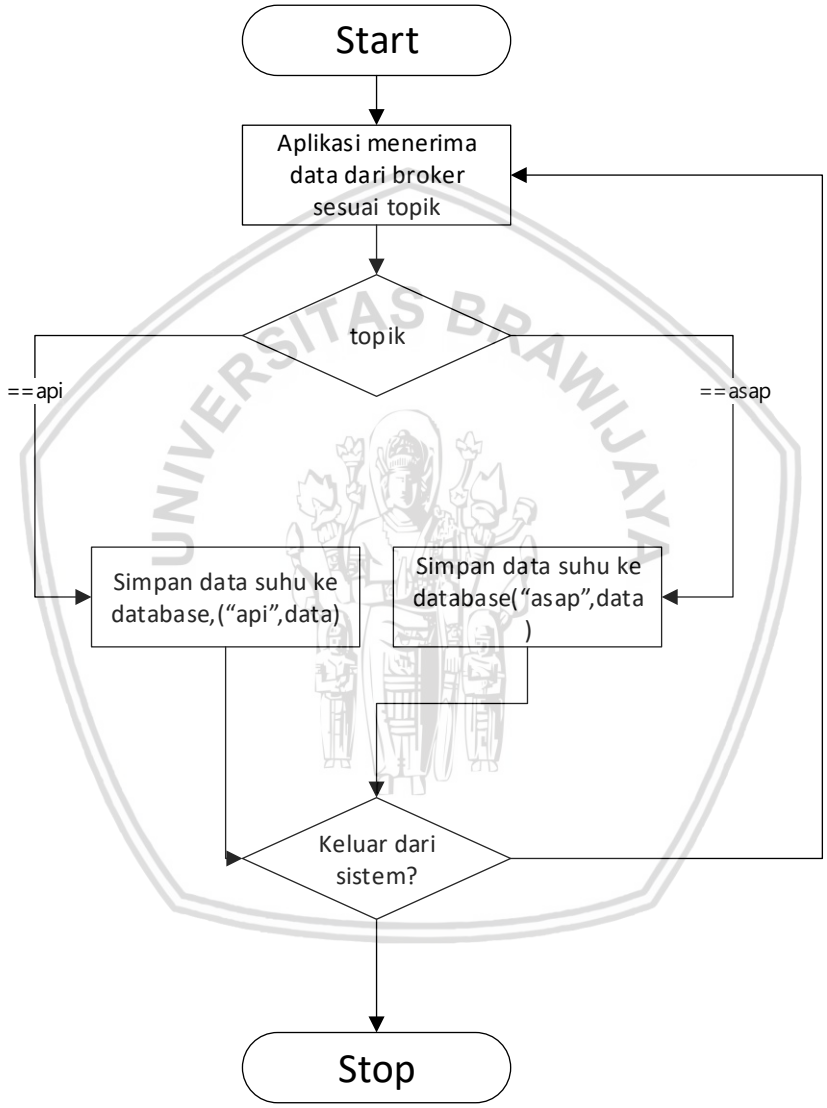
MQTT harus mempunyai topik dalam pengiriman maupun penerimaan data, jadi mikrokontroler yang tersambung dengan sensor MQ – 135 akan mengirim data seperti (asap,100) yang dimana asap merupakan topik dan 100 adalah nilai yang akan *dipublish* setelah itu data bisa dikirimkan menuju broker dan akan diteruskan kepada *subscriber* yang sudah melakukan *subscribe* dengan topik yang sama yaitu asap. Untuk proses pengiriman data pada sensor api dapat dilihat pada Gambar 4.11**:**



Gambar 4.11 Proses Pengiriman Data pada Sensor Api

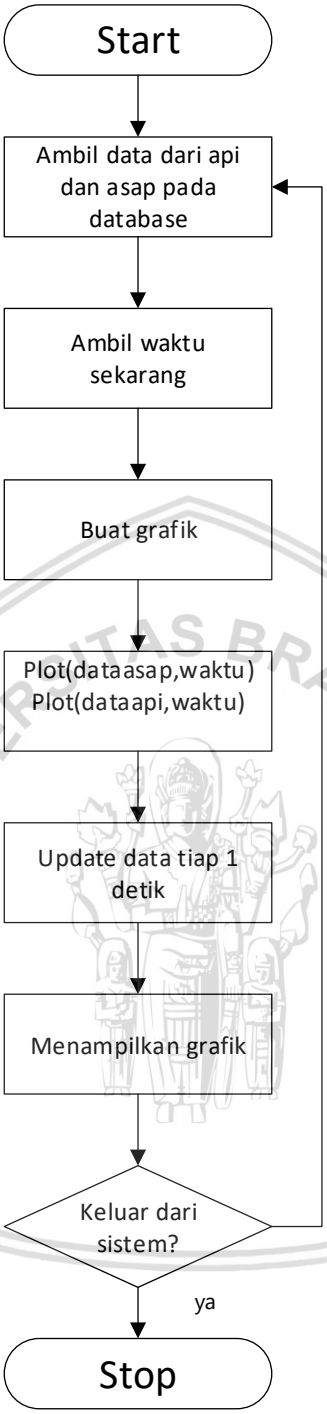
Sama seperti sensor MQ-135 pada sensor api ini melakukan perngiriman data setelah mebaca lingkungan dimana terjadi atau telihat api jika sensor menunjukan

adanya api maka akan meringimkan data menuju broker, berbeda dengan sebelumnya yang *mempublish* data dalam topik asap untuk *publish* kali ini menggunakan topik api dengan format yang sama (api,nilai) yang kemudian diterusakan kepada broker. Setelah semua data baik dari sensor api maupun MQ-135 telah *dipublish* maka aplikasi perangkat lunak sebagai *subscriber* akan mendapatkan data dari broker sesuai dengan topik yang telah *disubscribe* data tersebut itulah yang kemudian akan dikirimkan menuju database. skema pengiriman data pada databe dapat dilihat pada Gambar 4.12**:**



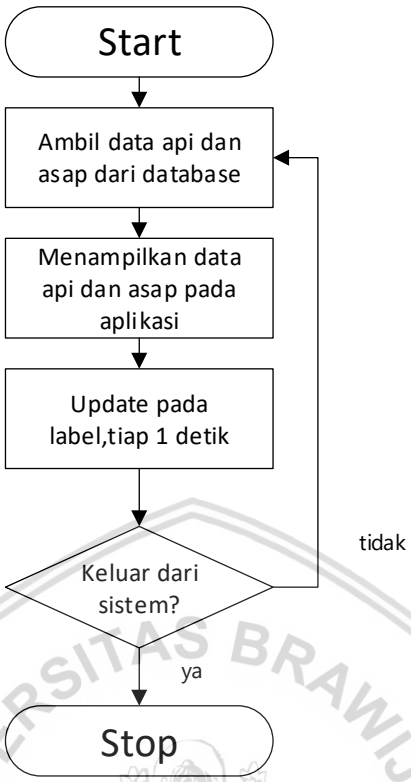
Gambar 4.12 Alur Penyimpanan Data pada Database

Setelah aplikasi perangkat lunak mendapatkan data dari *subscribe* pada topik api dan asap hal yang dilakukan oleh aplikasi ini adalah memasukan data tersebut pada database sehingga nantinya data yang ada pada database itulah yang akan dibuat untuk menampilkan data secara grafik ataupun data secara langsung pada Gambar 4.13 ini dapat dilihat alur bagaimana data ditampilkan secara grafik:



Gambar 4.13 Alur Menampilkan Data dalam Bentuk Grafik

Data dari sensor yang telah tersimpan pada database nantinya akan dipanggil lalu diubah menjadi *array* setelah itu ditambahkan dengan waktu sekarang dan selalu diupdate berkala sehingga didapatkanlah data dalam bentuk grafik secara realtime untuk memonitor hasil data dari sensor api dan sensor MQ-135

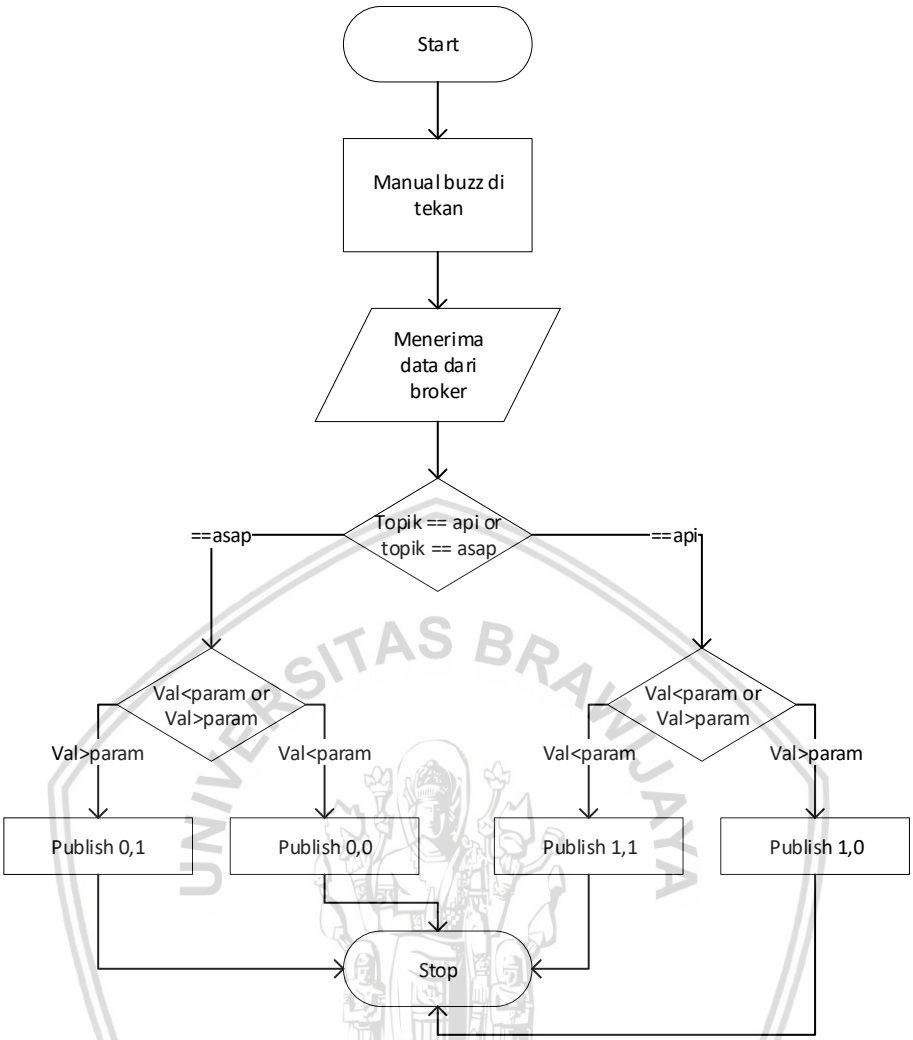


Gambar 4.14 Alur Menampilkan Data

Untuk menampilkan data *realtime* dari sensor juga sama dengan cara mengambil data dari database terlebih dahulu kemudian data tersebut diteruskan dan ditampilkan pada *container* yang ada diaplikasi jika tidak ada *container* akan berubah warna jika sensor melebihi parameter yang mengindikasikan adanya api atau asap.

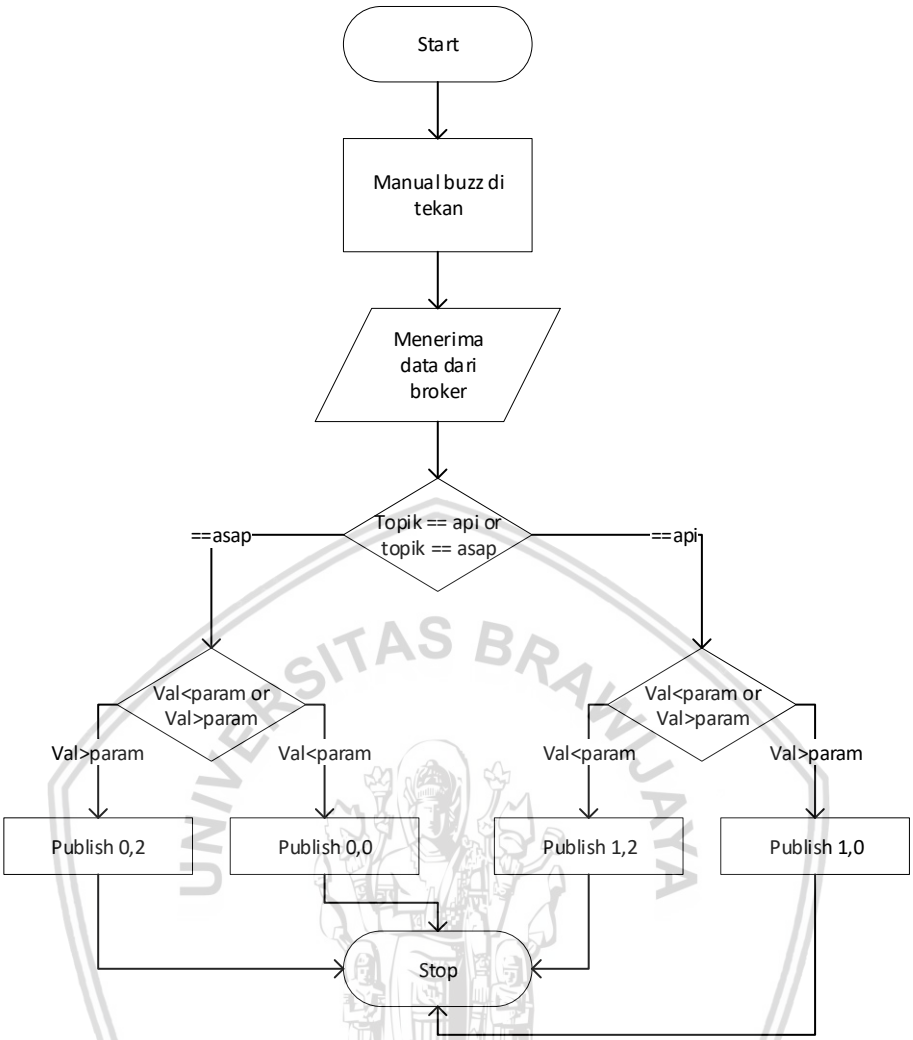
4.2.6 Perancangan Kontroling *Buzzer*

Perancangan ini untuk mengatur bagaimana kondisi *buzzer* dalam menjalankan berbagai perintah. Ada 2 perintah dalam perancangan ini yaitu secara otomatis dan manual dimana secara manual *buzzer* dimatikan dengan cara menekan tombol yang ada pada aplikasi sistem berdasarkan data sensor yang telah didapat jika data sensor berada diatas parameter maka *buzzer* harus bisa melakukan perintah pada Gambar 4.15 dapat dilihat alur *buzzer* dalam perintah manual:



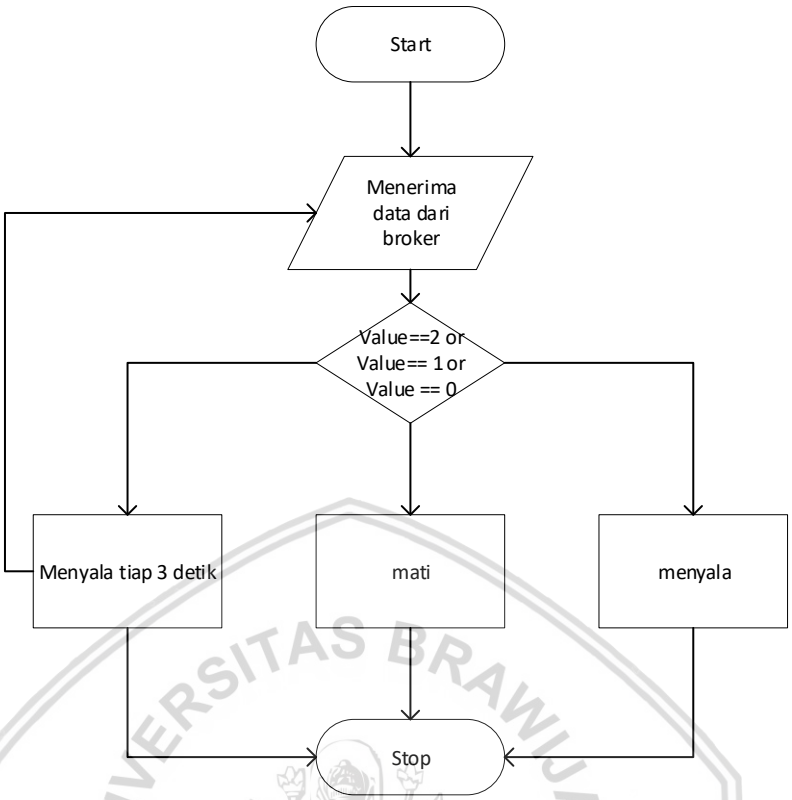
Gambar 4.15 Perancangan *Buzzer* secara Manual

Untuk perancangan *buzzer* yang pertama ini Perancangan *buzzer* secara manual yang dimana data yang sudah diterima akan disalurkan sesuai topik lalu jika parameter telah terpenuhi akan melakukan *publish* dengan nilai 1 untuk menyalakan *buzzer* dan 0 untuk mematikanya. Untuk Perancangan selanjutnya yaitu perancangan buzzer secara otomatis yang bisa dilihat pada gambar Gambar 4.16:



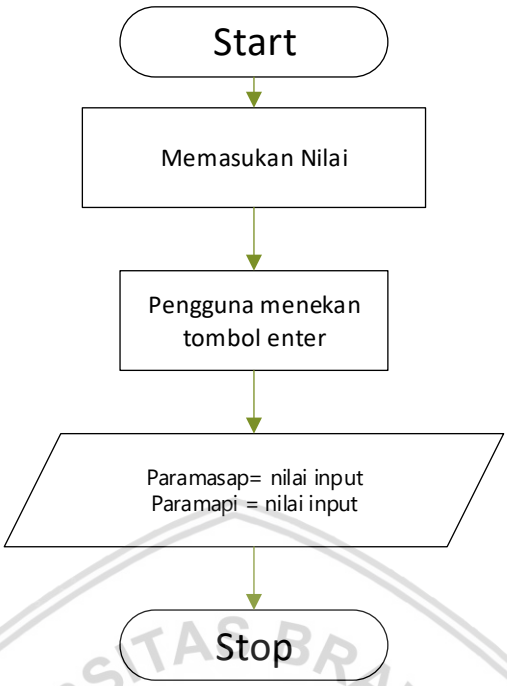
Gambar 4.16 Perancangan *Buzzer* secara Otomatis

Perbedaan antara Perancangan *buzzer* secara otomatis dan manual terletak pada koding yang dimana pada buzzer otomatis terdapat nilai yang bernilai 2 untuk menyalakan buzzer dengan interval tertentu dan menggunakan nilai 0 untuk mematikan *buzzer.* Dalam Mikrokontroler sendiri karena terhubung dengan *buzzer* secara langsung maka memiliki perencanaan dalam mematikan dan menghidupkan *buzzer* dapat di lihat dari Gambar 4.17**:**



Gambar 4.17 Perancangan Membunyikan dan Mematikan *Buzzer* pada Mikrokontroler

Pada perancangan sebelumnya jika mendapat nilai 1 menyebabkan *buzzer* menyala sedangkan nilai 0 membuat *buzzer* mati dan 2 menyebabkan *buzzer* menyala dengan interval tertentu. Untuk mengganti parameter pada sensor asap dan api dapat dilihat pada Gambar 4.18 :



Gambar 4.18 Rancangan Memasukan Nilai pada Parameter Asap dan Api

Pada Gambar 4.18 terlihat untuk menggati parameter hanya dengan memasukan nilai sesuai keinginan. Tujuan dari perancangan parameter ini adalah mengubah batasan dari sensor sehingga bisa diubah sesuai dengan kebutuhan.

BAB 5 IMPLEMENTASI

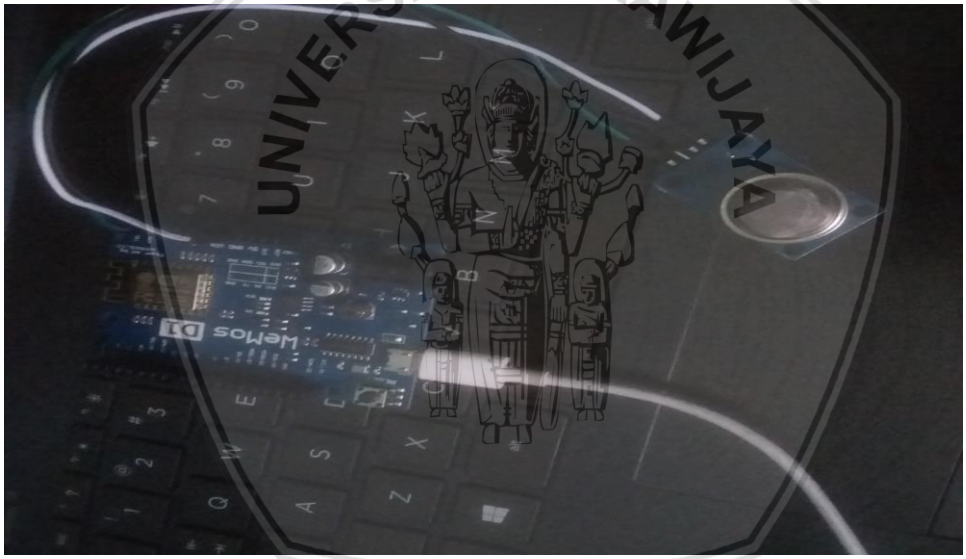
Untuk membuktikan jika perancangan kita berhasil salah satu parameternya adalah dengan dapat di terapkan pada tahap implementasi. Implementasi befokus lebih pada masalah teknis.

5.1 Implementasi Perangkat Keras

Menghubungkan antar perangkat agar dapat berkomunikasi adalah tujuan dari implementasi perangkat ini diharapkan semua perangkat dapat saling berkomunikasi antara sensor dengan mikrokontroler dan *buzzer* dengan mikrokontroler.

5.1.1 Implementasi Menghubungkan MQ-135 dengan Mikrokontroler

Untuk menghubungkan Mikrokontroler Wemos D1 R2 dengan sensor MQ- 135 dapat di lakukan dengan cara menghubungkan semua pin dari sensor menuju pin yang berada pada mikrokontroler. Proses ini dapat dilihat pada Gambar 5.1:

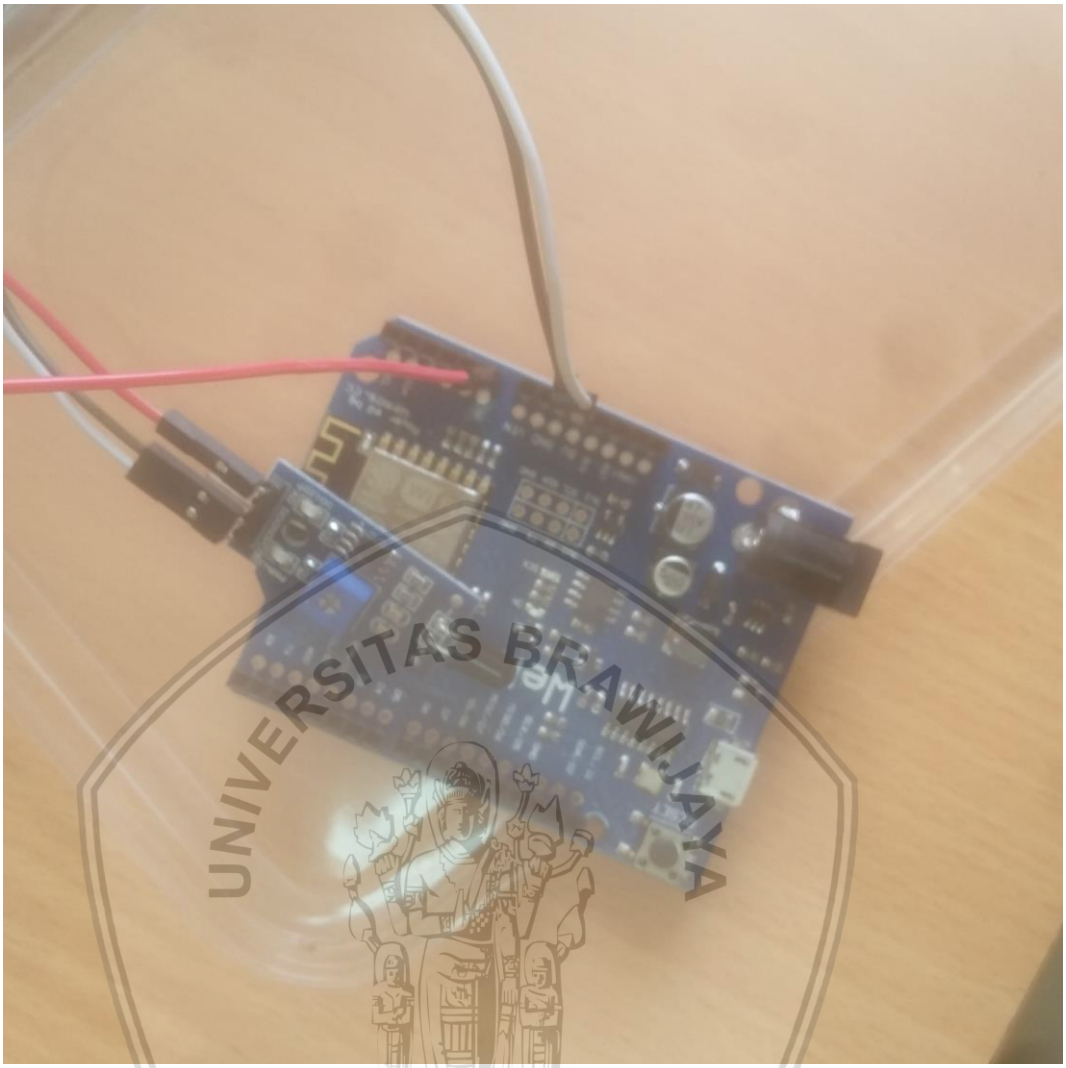


Gambar 5.1 Implementasi Menghubungkan MQ-135 dengan Mikrokontroler

Dengan bantuan kabel jumper berwarna putih pin Vcc pada sensor MQ-135 dengan pin 5v pada mikrokontroler sebagai sumberdaya, kemudian pin kedua pada sensor yaitu GND dihubungkan dengan pin GND pada mikrokontroler dengan bantuan kabel jumper warna hitam dan pin terahir pada sensor dihubungkan dengan pin A0 yang terdapat pada mikrokontroler sebagai analog output untuk menangkap data dari lingkungan.

5.1.2 Implementasi Menghubungkan Sensor Api dengan Mikrokontroler

Dikarenakan Sensor Api dan Sensor MQ-135 mempunyai *output* yang sama yaitu analog maka cara menghubungkan sensor dengan mikrokontroler juga tidak berbeda jauh. Pada Gambar 5.2 dapat dilihat hasil dari proses penghubungan:

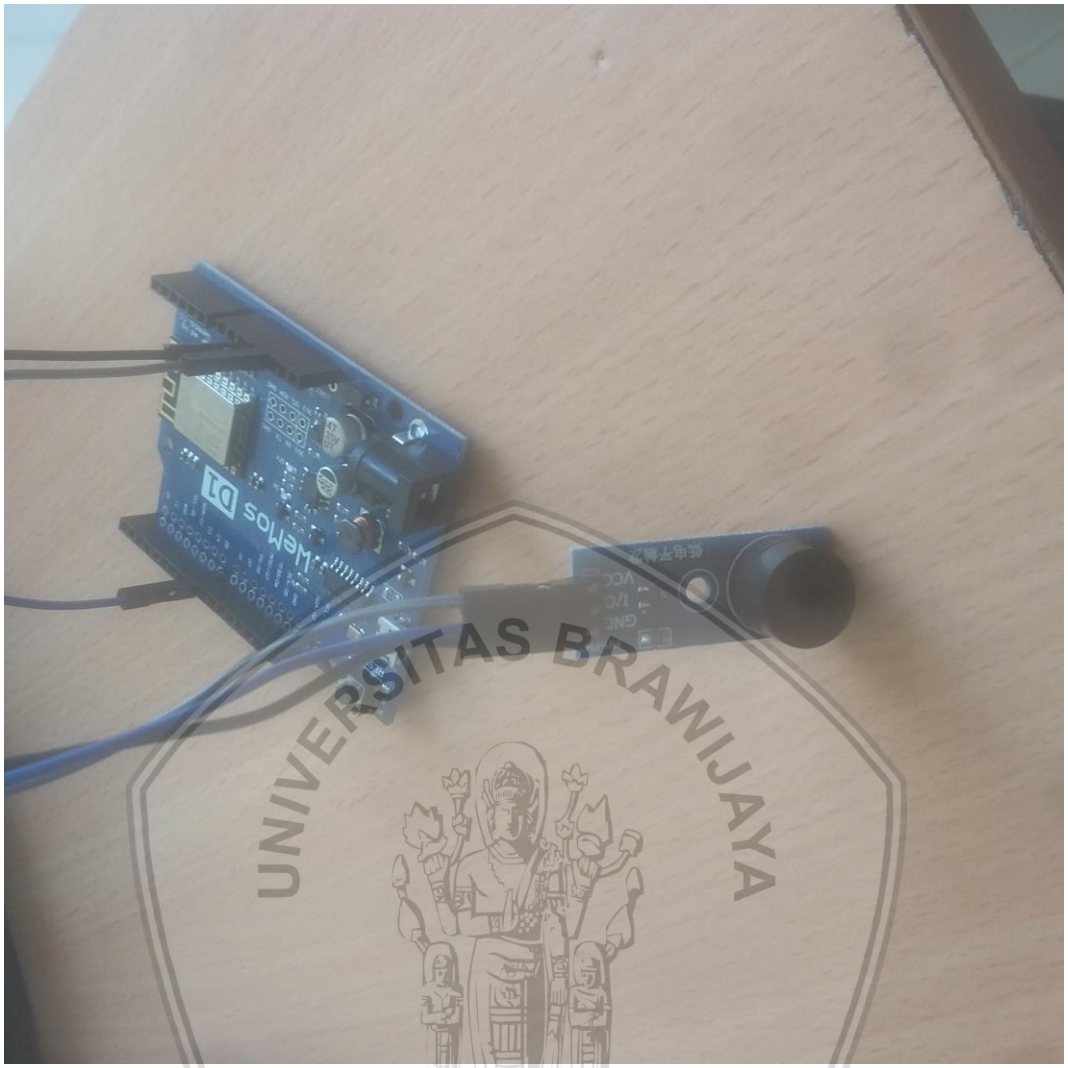


Gambar 5.2 Implementasi Integrasi SensorApi dengan Mikrokontroler

Menggunakan kabel jumper Pada **Gambar 5.2** kabel jumper berwarna putih menghubungkan pin Vcc sensor Api dengan pin 5V pada mikrokontroler sebagai sumber daya untuk sensor dan kabel jumper berwarna coklat menghubungkan pin sensor api dengan pin GND pada mikrokontroler wemos D1 R2. Meskipun sensor memiliki digital output tapi pada peneltiian ini menggunakan analog output yang terhubung melalui kabel jumper berwarna putih yang terhubung dengan pin A0 pada mikrokontroler.

5.1.3 Implementasi Integrasi *Buzzer* dengan Mikrokontroler

Buzzer memliki 3 pin, Vcc pin di hubungkan dengan pin 5v mikrokontroler sebagai sumber daya dengan bantuan kabel jumper warna abu abu. Pada pin kedua yaitu data yang bertipe digital jadi pin ini dihubungkan melalui kabel jumper berwarna biru menuju pin D Pada mikrokontroler dan pin terakhir adalah pin GND yang dihubungkan dengan pin GND pada mikrokontroler melalui kabel jumper berwarna hitam. Untuk hasil dari penghubungan sensor *buzzer* dengan mikrokontroler Wemos D1 R2 dapat dilihat pada Gambar 5.3 berikut:



Gambar 5.3 Implementasi Menghubungkan *Buzzer* dengan Mikrokontroler

5.1.4 Implementasi Arsitektur Jaringan

Dari 3 mikrokontroler yang sudah terintegrasi dengan modul ESP8266 salah satu akan dijadikan sebagai *access point* karena pada modul ESP8266 mempunyai fitur tersebut. Mikrokontroler yang terintegrasi dengan sensor MQ-135 akan menjadi *access point* tersebut pada Tabel 5.1 berikut menunjukan bagaimana membuat fungsi *access point* pada Wemos D1 R2:

Tabel 5.1 Implementasi Koneksi *Access Point*

```
1  IPAddress ip(192,168,4,113);
2  IPAddress gateway(192,168,4,1);
3  IPAddress subnet(255,255,255,0);
4  const char *ssid = "AP";
5  const char *password = "12345678";
6  WiFi.()
7  WiFi.softAP(ssid, password);
8  WiFi.softAPConfig(ip, gateway, subnet);
```

Pada penggalan Error! Reference source not found. penjelasan dari kode p rogram tersebut adalah sebagai berikut:

1. Pada baris 1- 3 membuat inisialisasi untuk alamat *access point* yang terdiri dari IP, *gateway* dan *subnet*
2. Pada baris 4-5 membuat inisialisasi variabel untuk nama dan password yang akan digunakan oleh akses point sehingga membentuk sebuah jaringan
3. Pada baris ke 7 adalah perintah untuk membuat akses point dengan parameter dari nilai variabel *ssid* dan *password*
4. Pada baris ke 8 inisialisasi alamat yang telah dibuat tadi kemudian dikonfigurasikan pada akses point sehingga memiliki alamat

Hal yang dilakukan ketika akses point sudah dibuat adalah menghubungkan semua perangkat lainya sehingga bisa saling berkomunikasi. Untuk melakukan hal tersebut maka dibuatlah mengakses jaringan yang telah dibuat dengan cara yang telah tertulis pada Tabel 5.2:

Tabel 5.2 Implemetasi Koneksi Mikrokontroler pada Akses Point

```
1  WiFi.begin(ssid, password);
2  WiFi.config(ip,gateway,subnet);
3  while (WiFi.waitForConnectResult() != WL_CONNECTED) {
4      Serial.println("Connection Failed! Rebooting...");
5      delay(5000);
6      Serial.print(".");
7      ESP.restart();
8    }
```

Penjelasan mengenai Tabel 5.2 adalah sebagai berikut:

1. Pada Baris 1 berfungsi untuk masuk dalam jaringan *access point* dengan nama dan *password* yang telah didefinisikan
2. Pada Baris ke 2 melakukan konfigurasi alamat secara statis
3. Pada Baris ke 3-8 merupakan perulangan dimana jika kita tidak bisa masuk pada jaringan yg kita inginkan akan dicoba kembali dalam 5 detik

5.1.5 Implementasi Koneksi Sistem dengan Mosquitto Brokker

Setelah semua perangkat keras sudah bisa berada dalam satu jaringan maka harus bisa berkomunikasi juga dengan broker Tabel 5.3 menjelaskan bagaimana cara berkomunikasi dengan broker:

Tabel 5.3 Implementasi Koneksi Mikrokontroler dengan Broker

```
1  const char* mqtt_server = "192.168.4.114";
2
3
4
5
6
7
```

```
8   client.setServer(mqtt_server, 1883);
9   while (!client.connected()) {
10      Serial.print("Attempting MQTT connection...");
11      if (client.connect("api")) {
12        Serial.println("connected");
13      } else {
          Serial.print("failed, rc=");
          Serial.print(client.state());
          Serial.println(" try again in 5 seconds");
          delay(5000);
      }
    }
```

Penjelasan lebih rinci Tabel 5.3 sebagai berikut:

1. Pada baris 1 merupakan inisialisasi alamat broker dengan ip sesuai dengan *ifconfig* atau bisa diubah menjadi *static*
2. Pada baris 2 adalah *setting port* yang akan digunakan oleh broker pada kode diatas menggunakan default port
3. Pada baris 3-13 merupakan perulangan yang terjadi saat tidak bisa terhubung dengan broker setiap 5 detik sekali akan mencoba untuk terhubung lagi
4. Baris 5 merupakan inisial untuk masuk ke broker

Untuk menghubungkan menghubungkan aplikasi perangkat lunak pada broker dapat dilihat pada Tabel 5.4 berikut:

Tabel 5.4 Implementasi Koneksi Aplikasi dengan Broker

```
1   Mqttc->connect
2
```

Bahasa pemograman yang digunakan dalam pembuatan aplikasi perangkat lunak adalah php penjelasan dari Tabel 5.4:

1. Memanggil fungsi MQTT yang yang ip dan *port* sudah tersimpan dalam database

5.1.6 Implementasi *Publish* atau *Subscribe* pada Sistem

Pembagian tugas subscribe dan *publish* adalah hal yang harus dilakukan setelah semua perangkat dapat terhubung dengan broker. Jika mikrokontroler mendapatkan tugas sebagai *publisher* maka dia harus mengirim data pada suatu topik tertentu dan hanya mikrokontroler yang terhubung dengan sensor saja yang bisa mengirim atau menerima data dari sensor mereka. Untuk *subscriber* akan menerima pesan dari data yang dikirimkan. Implementasi mikrokontroler yang terhubung dengan *buzzer* yang bertugas sebagai *subscriber* dapat dilihat pada Tabel 5.5 berikut:

Tabel 5.5 Implemtasi Peran *Subscribe* pada Mikrokontroler yang Terhubung dengan *Buzzer*

```
1  client.setCallback(callback);
2  client.subscribe("1");
3  client.subscribe("2");
4  void callback(char* topic, byte* payload, unsigned int length) {
5    Serial.print("Message arrived [");
6    Serial.print(topic);
7    Serial.print("] ");
8    for (int i = 0; i < length; i++) {
9      Serial.print((char)payload[i]);
10   }
11   Serial.println();
12
13   if (strcmp(topic,"1")==0) {
14     if((char)payload[0] == '1'){
15     tone(buzzer,1000);
16     Serial.println("buzzer menyala");
17     }
18     else{
19       noTone(buzzer);
20      Serial.println("buzzer mati");
21      }
22 }
23   if(strcmp(topic,"2")==0) {
24       if((char)payload[0] == '1')
25       {
26     tone(buzzer,1000);
27     Serial.println("buzzer menyala");}
28      else{
29     noTone(buzzer);
30     Serial.println("Buzzer mati");
31        }
32   }
33   }
```

Mikrokontroler yang terhubung dengan *buzzer* hanya akan menjalankan peran sebagai *subscriber* yaitu menerima pesan dari broker tanpa melakukan pengiriman pesan ke broker, penjelasan dari Tabel 5.5 adalah sebagai berikut:

1. Baris ke 1 menginisialisasi *method callback*, *method* callback adalah *method* yang akan dijalakan ketika *subscriber* menerima pesan dari broker.
2. Baris 2-3 untuk melakukan *subscriber* pada topik 1 dan topik 2.
3. Baris 4-33 adalah implementasi *method* callback yang berisi seleksi perintah untuk menyalakan dan mematikan *buzzer* yang terintegerasi dengan mikrokontroler.

Pada mikrokontroler yang lain peran mereka adalah sebagai *publisher*. Kode program untuk *publish* pada mikrokontroler yang terhubung dengan Sensor MQ-135 dapat dilihat pada Tabel 5.6 berikut:

Tabel 5.6 Implemtasi Peran *Publish* pada Mikrokontroler yang Terhubung dengan MQ-135

```
1  client.publish("asap",String(ppm).c_str());
```

Tugas dari sensor mikrokontroler yang terhubung dengan MQ-135 hanya menjalankan tugas sebagai *publisher* saja sehingga hanya mengirim data dari Sensor MQ-135 menuju broker, untuk penjelasan lebih detail dari Tabel 5.5 dapat dijabarkan sebagai berikut:

1. Pada baris 1 sensor mengirimkan pesan dengan topik asap lalu dengan nilai ppm yang diubah dalam bentuk *char string*

Tidak jauh beda sensor api juga memliki tugas yang sama seperti sensor MQ-135 dapat dilihat pada Tabel 5.7 berikut:

Tabel 5.7 Implemtasi Peran *Publish* pada Mikrokontroler yang Terhubung dengan Sensor Api

```
1   client.publish("api",fire);
```

penjelasan untuk sepotong dari Tabel 5.7 adalah:

1. Pada baris 1 sensor mengirimkan pesan dengan topik api lalu dengan nilai sensor yang diubah dalam bentuk *char string*.

Pada aplikasi rekayasa perangkat lunak untuk *subscribe* dapat dilihat pada Tabel 5.8 berikut:

Tabel 5.8 Implemtasi Peran Publish atau Subscribe pada Aplikasi

```
1   msgFlame = subscribe.simple(topicFlame,hostname=mqhost,port=mqport)
2           #msgSmoke =
3   subscribe.simple(topicSmoke,hostname=mqhost,port=mqport)
4           print("%s %s" % (msgFlame.topic, msgFlame.payload))
5           #print("%s %s" % (msgSmoke.topic, msgSmoke.payload))
6
7           if(not(msgFlame.payload is None)):
8                   addr =
9   'http://localhost/guimqtt/php/receive.php?type=0&val=' +
10  str(msgFlame.payload)
11                  resp = urllib.urlopen(addr).read()
12                  print resp
13          '''
14          if(not(msgSmoke.payload is None)):
15                  addr =
16  'http://localhost/guimqtt/php/receive.php?type=1&val=' +
17  str(msgSmoke.payload)
18                  resp = urllib.urlopen(addr).read()
                    print resp
            '''
```

tanda # adalah dan ''' adalah *string* yang diabaikan tapi disini digunakan untuk *flame* karena sama. Penjelasan untuk Tabel 5.8 adalah sebagai berikut:

1. Pada baris 1 melakukan *subscribe* pada topik *Flame* beserta alamat brokernya
2. Pada Baris ke 3-4 menampilkan *payload* yang didapat dari hasil *subscribe* dalam konsol
3. Pada baris 6-7 akan menampilkan data *payload* diaplikasi

5.2 Implementasi Perangkat Lunak

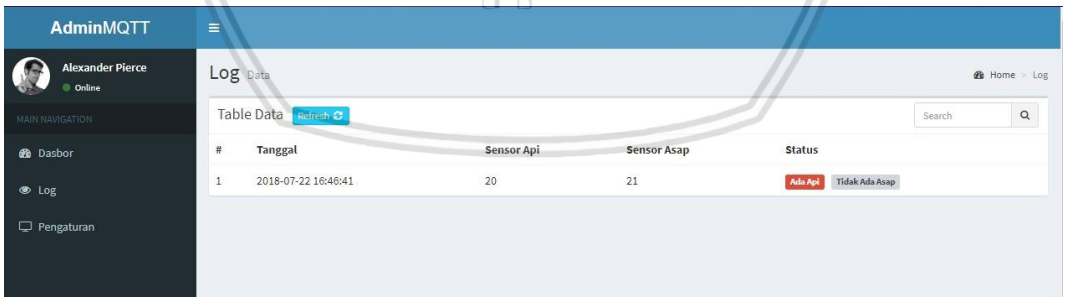
Merupakan Implementasi dimana perangkat lunak dibuat. Fungsi dari perangkat lunak tersebut apakah sudah terpenuhi.

5.2.1 Implementasi Aplikasi Sistem Kendali

Aplikasi peranagkat lunak ini dibuat bertujuan untuk menbantu pengguna dalam memonitor dan mengkontrol dan memiliki fitur tertentu dilihat pada Gambar 4.7**,** Gambar 4.8, dan Gambar 4.9 berikut:

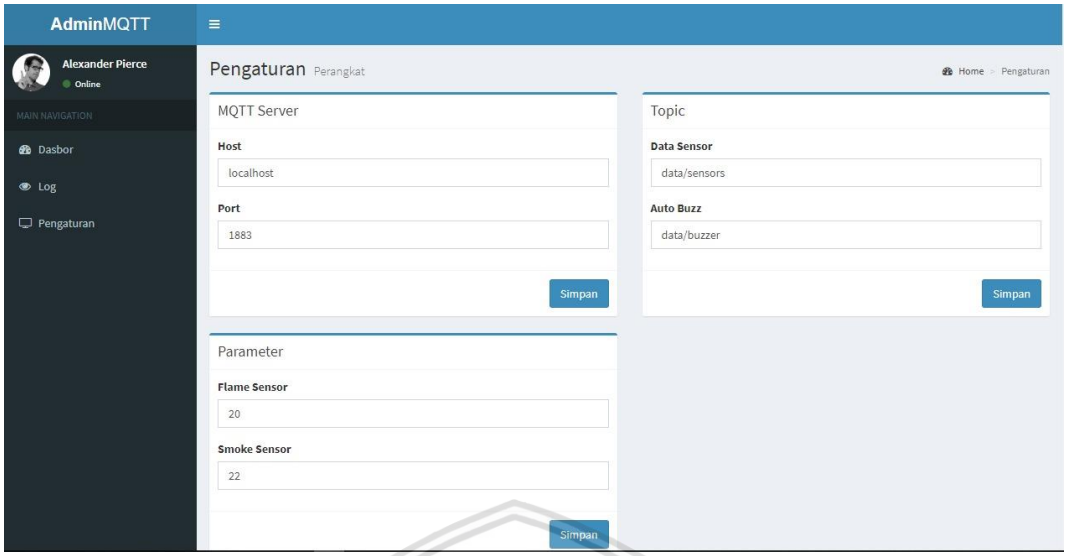
Gambar 5.4 Implementasi Aplikasi Menu Dashboard

Pada Gambar 5.4 terlihat data sensor, grafik dan kontorling *buzzer* bertujuan untuk memudahkan pengguna dalam mencerna data tangkapan sensor serta mengatur *buzzer*.



Gambar 5.5 Implementasi Aplikasi Menu Log

Pada Gambar 5.5 penguna dapat melihat riwayat data sensor dan status dari sensor tersebut serta nilai data yang tertangkap oleh sensor.



Gambar 5.6 Implementasi Aplikasi Menu Pengaturan

pusat kontrol dari aplikasi untuk mengubah nilai dari parameter untuk sensor serta pengaturan MQTT dan topik.

5.2.2 Implementasi Database

Menggunakan databe MySQL dengan nama dbmqtt untuk menyimpan nilai baik dari sensor maupun dari sistem.

Tabel 5.9 Implementasi Database

```
1  $conn = mysqli_connect("localhost","root","","dbmqtt");
2
```

Penjelasan Tabel 5.9 Implementasi Database adalah:

1. Baris 1 pada progam adalah perintah untuk membuat konkesi dengan database yang berada dalam *localhost username root* dan nama databasenya adalah dbmqtt

5.2.3 Implementasi Monitoring Asap dan Api dari Sensor

Tujuan dari monitoring asap dan api ini adalah agar pengguna bisa tau serta mendapatkan data dari lingkungan untuk mencegah terjadinya kebakaran dengan cara dapat melakukan tindakan secepatnya. proses pengiriman data untuk monitoring dapat dijelaskan pada Tabel 5.10**:**

Tabel 5.10 Implementasi Pengiriman Data Asap ke Broker

```
1  float ppm = gasSensor.getPPM();
2    Serial.println(ppm);
3    client.publish(asap, String(ppm).c_str());
4    delay(1000);
5
6
```

Pada Tabel 5.10 menunjukan kode dari mikrokontroler dan sensor MQ-135 yang kemudian akan diteruskan menuju broker, penjelasan lebih rinci Tabel 5.10 adalah sebagai berikut:

1. Pada baris 1 data dari lingkungan diinisialisasikan pada sensor.
2. Padabaris 2 untuk menampilkan nilai dari sensor pada konsol
3. pada baris 3 *publish* data ppm yang telah diubah terlebih dahulu menjadi *char string*
4. Pada baris 4 membatasi pengiriman data pada broker setiap 1 detik sekali

Tabel 5.11 Implementasi Pengiriman Data Api ke Broker

```
1    sensor = analogRead(fire);
2    Serial.println(sensor);
3    client.publish(api, String(sensor).c_str());
4   delay (1000);
5
6
```

Pada Tabel 5.11 menunjukan kode dari mikrokontroler dan sensor api yang digunakan untuk mengambil data dari sensor api kemudian akan diteruskan menuju broker, penjelasan lebih rinci dari Tabel 5.11 adalah:

1. Pada baris satu memberi nilai pada sensor beru data yg diambil oleh sensor api dalam (*fire*)
2. Pada baris 2 mencetak nilai sensor pada konsol
3. Pada baris 3 *publish* data menuju broker dengan merubah nilai sensor menjadi *char string*
4. Pada baris 4 pengiriman pada broker di atur setiap 1 detik sekali

Setelah berhasil mengirim data ke broker kemudian data akan diteruskan ke database melalui aplikasi perangkat lunak, data dari databse inilah yang akan digunakan untuk membuat tampilan dalam bentuk grafik runtutan dalam menampilkan grafik dapat dilihat pada Tabel 5.12 berikut:

Tabel 5.12 Implementasi Menampilkan Data pada Grafik Api

```
1   name: 'FLAME',
2                           data: [
3                               <?php
4                               $cn = 0;
5                               foreach($api as &$value){
6                                   $cn++;
7                               }
8                               if($cn<=7){
9                                   foreach($api as &$value){
10
```

```
                                        echo $value;
                                        echo ",";
                                    }
                                }
                                else{
                                    for($ver=0;$ver<=7;$ver++){
                                        echo $api[$ver].",";
```

Penjelasan tentang Tabel 5.12 diatas dapat dijelaskan sebagai berikut:

1. Pada baris 2-16 data api yg didapat akan terus ditapilkan pada grafik beserta dengan nilainya

Tabel 5.13 Implementasi Menampilkan Data pada Grafik Asap

```
{
                            name: 'SMOKE',
                            data: [
                                <?php
                                $cn = 0;
                                foreach($asap as &$value){
                                    $cn++;
                                }
                                if($cn<=7){
                                    foreach($asap as &$value){
                                        echo $value;
                                        echo ",";
                                    }
                                }
                                else{
                                    for($ver=0;$ver<=7;$ver++){
                                        echo $asap[$ver].",";
```

Sama seperti pada penampilan sensor asap penjelasan tentang Tabel 5.13 diatas dapat dijelaskan sebagai berikut:

1. Pada baris 2-16 data asap yang didapat akan terus ditampilkan pada grafik beserta dengan nilainya

Setelah menampilkan data pada grafik sesuai dengan perancangan yang dibuat, selanjutnya dilakukan implementasi menampilkan data asap dan api pada label didalam aplikasi, implementasi untuk menampilkan data pada label aplikasi dapat dilihat pada Tabel 5.14 berikut:

Tabel 5.14 Implementasi Menampilkan Data pada Aplikasi

```

            if($val == 'flame'){
        $sql = mysqli_query($conn,"SELECT * FROM datares WHERE
Type=0 ORDER BY Id DESC LIMIT 1");
        $data = mysqli_fetch_array($sql);

        $title = 'Sensor Api';
        $state = 'Tidak Ada Api';
        if($data['Status'] == 1){
            $bg = 'bg-red';
```

```
11              $state = 'Ada Api';
12          }
13          $_val = $data['Value'];
14          $show = true;
15      }
16      else if($val == 'smoke'){
17          $sql = mysqli_query($conn,"SELECT * FROM datares WHERE
18  Type=1 ORDER BY Id DESC LIMIT 1");
19          $data = mysqli_fetch_array($sql);
20
21          $title = 'Sensor Asap';
22          $state = 'Tidak Ada Asap';
23          if($data['Status'] == 1){
24              $bg = 'bg-red';
25              $state = 'Ada Asap';
26          }
27          $_val = $data['Value'];
28          $show = true;
```

Penjelasan dari Tabel 5.14 adalah sebagai berikut:

1. Baris 1-2 digunakan untuk merubah *text* pada label *textbox* dengan nilai data sensor yang diambil dari database, *script* ini terdapat pada *method update*.
2. Baris 3 digunakan perintah untuk menjalakan *method update* setiap 1000ms agar data yang ditampilkan *realtime*.

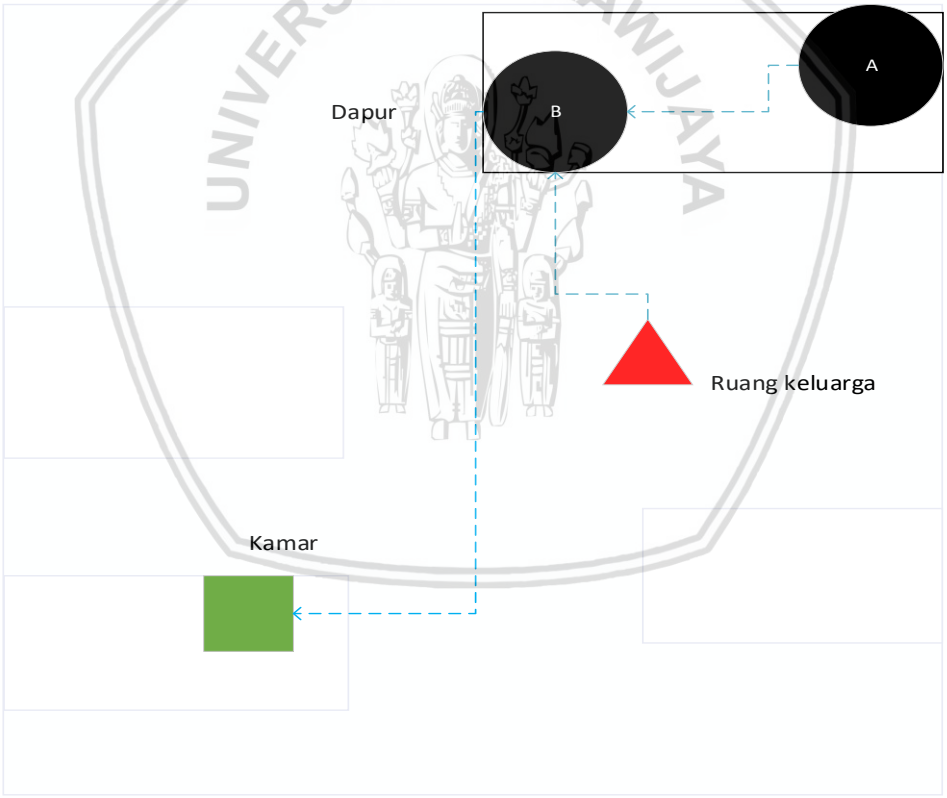
BAB 6 PENGUJIAN DAN ANALISA HASIL PENGUJIAN

6.1 Perancangan Pengujian

Perancangan pengujian digunakan untuk Mengetahui bagaimana aplikasi sudah sesuai perancangan dan menguji apakah dapat menjalankan protokol MQTT serta menguji kinerja sistem dan pengujian fungsionalitas dari sistem yang telah dibuat.

6.1.1 Perancangan Pengujian Sistem

Pengujian sistem dibuat untuk mengetahui apakah sistem sudah menerapkan protokol komunikasi dengan baik. Protokol komunikasi yang digunakan adalah MQTT diharapkan dengan menggunakan protokol MQTT pertukaran data dan komunikasi antar perangkat dapat berjalan, dengan cara mengukur seberapa cepat reaksi sistem dari sensor sampai *buzzer* bunyi.



Gambar 6.1 Perancangan Pengujian

Lingkaran Hitam bernama B adalah Sensor MQ-135 yang terhubung dengan mikrokontroler Wemos D1 R2 dan menjadi *access point* pada konfigurasi ESP 8266. Semua perangkat harus terhubung terlebih dahulu secara *wireless* seperti yang ditunjukan pada gambar Gambar 6.1, Kotak hijau adalah Laptop sebagai broker dan aplikasi perangkat lunak, lalu segitiga merah adalah buzzer yang sudah terhubung dengan mikrokontroler dan lingkaran A saling terhubung melalui

Mikrokontroler dari sensor B. Setelah terhubung sensor kemudian mengirimkan data pada broker yang terletak pada laptop dan diteruskan ke aplikasi, pada tahap ini terjadi pertukaran data menggunakan protokol MQTT jika data yang dikirim melebihi parameter yang ada maka akan membuat *buzzer* berbunyi. Setelah protokol MQTT sudah bisa dilakukan maka dapat diukur kinerja sistem dengan cara membandingkan waktu yang dibutuhkan sistem untuk dapat membuat buzzer berbunyi.

6.1.2 Perancangan Pengujian Fungsionalitas

Untuk mengetahui bahwa fitur yang dibuat dalam perancangan dapat dilakukan dengan baik maka dibutuhkan perancangan fungsionalitas sistem, skenario yang dapat dilakukan antara lain:

Tabel 6.1 Skenario Pengujian Menyalakan *Buzzer* dengan Tombol Manual

Test Case	Pengujian tombol manual button
Prosedur	menekan tombol manual yang berada pada menu dasbor
Hasil yang di inginkan	*Buzzer* mati

Tabel 6.2 Skenario Pengujian Mematikan *Buzzer* dengan Tombol Manual

Test Case	Pengujian tombol manual button
Prosedur	menekan tombol manual yang berada pada menu dasbor
Hasil yang di inginkan	*Buzzer* Menyala

Tabel 6.3 Skenario Pengujian *Buzzer* Auto

Test Case	Pengujian menyalakan dan mematikan *buzzer* secara otomatis
Prosedur	menekan tombol auto pada menu dasbor

Hasil yang yang diinginkan	*Buzzer* mati dan menyala tergantung dengan hasil sensor

Tabel 6.4 Skenario Pengujian Menampilkan Grafik

Test Case	Pengujian menampilkan grafik pada aplikasi
Prosedur	Jalankan aplikasi dan maintest
Hasil yang yang diinginkan	Data dapat ditampilkan dalam bentuk grafik

Tabel 6.5 Merubah Parameter Nilai Sensor

Test Case	Merubah parameter sensor
Prosedur	1.Mengsi nilai pada box 2. *save*
Hasil yang yang diinginkan	Parameter berubah sesuai dengan inputan

Tabel 6.6 Menampilkan Data

Test Case	Menampilkan nilai data pada aplikasi
Prosedur	Jalanakan aplikasi
Hasil yang yang diinginkan	Dapat menampilkan nilai sesuai dengan data yang dikirim

Tabel 6.7 Menampilkan Riwayat Data

Test Case	Menampilkan riwayat data sensor
Prosedur	1.Jalankan aplikasi 2.Buka menu Log
Hasil yang yang diinginkan	Dapat menampilkan nilai sesuai dengan data yang sudah disimpan

6.2 Hasil dan Analisis Pengujian

Tahap ahir dari penilitian ditujukan untuk mengetahui jika sistem telah dibuat sesuai dengan perancangan dan tujuan.

6.2.1 Hasil Pengujian Sistem

Pengujian sistem dilakukan dengan cara menghitung berapa waktu yang dibutuhkan sistem untuk merespon jika kita memberi *trigger* sehingga buzzer *bunyi*.

Tabel 6.8 Waktu Tempuh Sensor

No	sensor	Waktu
1	api	1 sec
2	api	1 sec
3	api	1 sec
4	asap	1 sec
5	asap	1 sec
6	asap	1 sec
7	api dan asap	1 sec
8	api dan asap	1 sec
9	api dan asap	1 sec

Pada Tabel 6.8 waktu yang di butuhkan sistem untuk merespon masukan dari sensor adalah 1 detik, berdasarkan pengujian di atas, maka dapat disimpulkan bahwa sistem bisa cukup cepat untuk dapat mendeteksi indikasi adanya kebakaran.

6.2.2 Hasil Pengujian Fungsionalitas

Hasil dan analisis dari pengujian fungsionalitas dari sistem harus sesuai dengan perancangan pengujian yang dibuat, sehingga fungsi yang telah dibuat sesuai tujuan awal:

Tabel 6.9 Hasil dan Analisis Pengujian Fungsionalitas Aplikasi

No	Deskripsi	Masukan	Keluaran yang Diharapkan	Hasil yang Didapat	Kesimpulan
1.	Pengujian Mematikan *Buzzer* secara manual	Pengguna menggeser tombol pada manual pada aplikasi	• *Buzzer* mati.	• *Buzzer* mati.	Benar
2.	Pengujian Menyalakan *Buzzer* secara manual	Pengguna menggeser tombol pada manual pada aplikasi	• *Buzzer* bunyi	• *Buzzer* bunyi	Benar
3.	menyalakan dan mematikan *Buzzer* secara otomatis	Pengguna menggeser tombol auto pada aplikasi	• *Buzzer* bunyi dan mati sesuai dengan parameter dari sensor	• Sesuai dengan parameter sensor	Benar
4.	Pengujian menampilkan grafik pada aplikasi	Menjalankan aplikasi	• Nilai grafik sesuai dengan nilai pada database MySQL	• Sama	Benar
5.	Pengujian merubah nilai parameter asap dan api	Input nilai	• dapat merubah nilai parameter asap dan api sesuai dengan nilai yang diinputkan.	• nilai berubah sesuai inputan	Benar
6.	Pengujian menampilkan nilai data sensor pada aplikasi	Menjalankan aplikasi	• dapat menampilkan nilai terahir sensor	• nilai terahir sensor muncul	Benar

7	Pengujian menampilkan riwayat data sensor	Membuka menu Log	• dapat menampilkan 50 data terahir	• data dapat di tampilkan	Benar

Dari Tabel 6.9 semua skenario pengujian dapat dilakukan dengan benar sesuai dengan keluaran yang diharapkan, disamping itu berjalanya sistem dengan baik tidak lepas pula dari terhubungnya semua sistem dan berjalanya protokol MQTT dalam sistem. Hal ini dapat di buktikan pada percobaan 1 – 3, untuk menyalakan buzzer sistem harus bisa mengimplementasikan *publish* dan *subscribe* terlebih dahulu yang merupakan cara protokol MQTT berkomunikasi. Dengan subscribe buzzer akan menyala sesui dengan topik yang *disubscribe* jika mendapat data dari topik auto maka dia akan menyala sesuai dengan durasi yang telah ditetapkan yaitu 3 detik, sedangkan jika mendapat data dari topik manual maka *buzzer* akan terus berbunyi. Hal ini tidak terlepas dari peran broker Mosquitto sebagai jembatan dalam pengriman data pada Percobaan 4 dan 5 peran broker dalam *publish* data sangat besar, oleh karena itu keberhasilan dalam menampilkan data ini merupakan tolak ukur bahwa protokol MQTT telah berhasil dijalankan dalam sistem aplikasi ini. Pada percobaan 5 merupakan penerapan kontroling pada sistem dengan dapat merubah data inputan menyebabkan sensor bereaksi terhadap hal tersebut. Pada percobaan 6 di dapatkan rekap dari data sensor telah dikirim hal ini membuktikan bahwa nilai data yg dikirim dan disimpan adalah sama.

BAB 7 KESIMPULAN DAN SARAN

7.1 Kesimpulan

Hasil dari perancangan, implementasi, pengujian, serta analisis dan hasil pengujian dari Implementasi Sistem Kebakaran dapat disimpulkan:

1. Semua perangkat dapat terhubung dalam *access point* dari mikrokontroler sehingga semua perangkat dapat berkomunikasi.
2. Protokol MQTT dapat di terapkan pada semua perangkat yang terhubung pada aplikasi. Protokol MQTT digunakan dalam proses pertukaran data dengan menggunakan *publish subcribe* berdasarkan topik yang telah dipilih.
3. *Publisher* yaitu mikrokontroler dapat mengubah kondisi *subscriber buzzer* agar dapat melakukan perintah seperti bunyi atau mati dari besar nilai data yang dikirimkan.
4. Berdasarkan hasil dari pengujian fungsional dan sistem, sistem yang telah dibuat mampu menjalankan semua fungsionalitas dan dari semua sensor waktu yang dibutuhkan oleh sistem untuk mengirim data adalah 1 detik.

7.2 Saran

Setelah selesai melakukan penelitian ada sedikit saran yang disampaikan penulis utuk mengembangkan penelitian ini:

1. Perlu penambahan pada sensor sehingga cara pendeteksian lebih banyak.
2. Perlu penambahan dalam penilaian indikasi secara lebih detail.
3. Perlu dilakukan lagi penelitian menggunakan mikrokontroler yang lain.

DAFTAR PUSTAKA

Akarsh Goyal, I. K. A. K. M., 2016. PDConnect: Low Cost Wearable Device connecting Patient-Doctor via Cloud for Good Health.

Anusha.M, S. B. S. M. R. V. K. B., 2017. PERFORMANCE ANALYSIS OF DATA PROTOCOLS OF INTERNET OF THINGS: A QUALITATIVE REVIEW. Volume 115.

Eclipse, 2011. *Eclipse Paho.* [Online] Available at: https://eclipse.org/paho/ [Accessed 30 7 2018].

Eclipse, 2013. *Mosquitto.* [Online] Available at: https://www.eclipse.org/proposals/technology.mosquitto/ [Accessed 30 8 2018].

Himanshu Singh, V. P. V. K. a. V. U., 2018. IoT based Smart Home Automation System using Sensor Node.

Jakarta, B., 2016. *Indeks Berita Bencana: Kebakaran.* [Online] Available at: https://bpbd.jakarta.go.id/news/category/kbr [Accessed monday July 2018].

Lampkin, V. et al., 2012. *Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry.* 1st ed. USA: WebSphere MQ.

Seong-Min Kim, H.-S. C. W.-S. R., 2015. IoT Home Gateway for Auto-Configuration and Management of MQTT Devices.

Solace, n.d. *MQTT Topics.* [Online] Available at: http://docs.solace.com/Features/MQTT-Topics.htm [Accessed 30 8 2018].

sparkfun, n.d. *WiFi Module - ESP8266.* [Online] Available at: https://www.sparkfun.com/products/13678 [Accessed 30 8 2018].

Wemos, n.d. *Wemos D1.* [Online] Available at: https://www.wemos.cc/product/d1.html [Accessed 30 8 2018].